TU Kaiserslautern

Solving Integer Programs using the Algorithm of Hosten and Sturmfels

Bachelor Thesis in Mathematics

Author
Sebastian Muskalla

Supervisor
Thomas Markwig

July 15, 2013

Contents

1 Introduction

The goal of this bachelor thesis is to show how one can translate certain integer linear optimization problems into the language of multivariate polynomials and solve them using methods from computer algebra.

In the second section I will define the basic notions from optimization, which we will need to describe the integer programs we look at. I will show how to bring arbitrary integer programs into a certain form which we will use later to translate them into algebraic terms. Furthermore, I will cite some results on the complexity of optimization problems, which will show that solving integer programs efficiently is a task of interest in modern mathematics.

In the third section we will look at the algorithms for reduction and computing Gröbner bases, which we will need to solve the optimization problem. We will also see how we can use the cost function of a linear program to define a monomial ordering. This gives us the first step in the way to translating our problem.

In the fourth section we will associate the toric ideal I_A to the constraint matrix A of an integer program and see that we can use this ideal to reduce an initial solution of the integer program to an optimal one, using the algorithms introduced in the previous section.

The only problem that we are left with is the computation of a Gröbner basis of I_A. We will show in the fifth section general properties of I_A which would allow to compute a Gröbner basis in various ways. Several of the algorithms for different types of integer programs are described in the diploma thesis of Christine Theis [The99], on which this bachelor thesis is based.

We will study one of these algorithms in detail, which was introduced by Serkan Hosten and Bernd Sturmfels in [HS95].

2 Basics of Linear and Integer Optimization

In this section, we will define the basic structure of the problems we will later try to solve. Therefore, I will introduce linear and integer programs and show that we can bring them into a so called standard form, which will allow us to treat them without case-by-case analysis.

2.1 Definition: Linear Program

A **linear program (LP)** is a problem of the following form:

Find a vector $x = (x_1, ..., x_n)^T \in \mathbb{R}^n$ (for some $n \in \mathbb{N}, n > 0, n < \infty$) subject to

- a finite amount of equal-constraints of the form:
  $a_{i1} * x_{i1} + ... + a_{in} * x_{in} = b_i$ for some $i \in \{1, ..., i_{max}\}$,
- a finite amount of less-or-equal-constraints of the form:
  $a_{j1} * x_{j1} + ... + a_{jn} * x_{jn} \leq b_j$ for some $j \in \{i_{max} + 1, .., j_{max}\}$,
- a finite amount of greater-or-equal-constraints of the form:
  $a_{k1} * x_{k1} + ... + a_{kn} * x_{kn} \geq b_k$ for some $k \in \{j_{max} + 1, .., k_{max} = m\}$,

  where for all $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$: $a_{ij} \in \mathbb{R}$, $b_j \in \mathbb{R}$

- and some sign-constraints of the form:
  $x_i \geq 0$ for $i \in S$
  where $S \subseteq \{1, ..., n\}$

such that the objective function value
$c_1 * x_1 + ... + c_n * x_n$
for some $c_i \in \mathbb{R}$, $i \in \{1, ..., n\}$ is minimal (respectively maximal).

2.2 Definition: Integer Program

We call a problem an **integer program (IP)** if it has the same form as described in definition 2.1, but we only search for integer solution vectors $x \in \mathbb{Z}^n$ and all occurring coefficients in the constraints are integer: $a_{ij} \in \mathbb{Z}$ and $b_j \in \mathbb{Z}$ (for all $i \in \{1, ..., n\}$ and all $j \in \{1, ..., m\}$).

2.3 Example

Consider the following problem:
Suppose an automaker wants to transport 50 new cars from its production facility to the city where they should be sold. It can hire a company C_1 to transport them for \$100 per car or a company C_2 to transport them for \$80 per car. Due to limited resources, each company cannot transport more than 40 cars. How should the automaker split the

cars among the companies such that the expense is minimal?

This problem can be formulated as integer program as follows:
Find $x = (x_1, x_2)^T \in \mathbb{Z}^2$ subject to the constraints
$x_1 \leq 40$,
$x_2 \leq 40$,
$x_1 + x_2 = 50$,
$x_1, x_2 \geq 0$,
such that $100 * x_1 + 80 * x_2$ is minimal,
where x_1 corresponds to the number of cars transported by the company C_1 and x_2 to the number of cars transported by company C_2.

Note that $v' = (40, 10)^T$ is a feasible solution of the integer program with objective function value $100 * 40 + 80 * 10 = 4800$.

2.4 Definition
Suppose we have a linear program (LP).

We call the components $x_i \in \mathbb{R}$ of the vector $x \in \mathbb{R}^n$ we are searching for the **variables** of (LP).

We call a vector $x \in \mathbb{R}^n$ a **feasible solution** for (LP) if the vector fulfills all constraints of (LP).

For any vector $x \in \mathbb{R}^n$ exactly one of following properties is true:

- It may not be a solution of (LP): the vector is not a feasible solution of (LP).
- It may be a feasible optimal solution of (LP): the vector is a feasible solution of (LP) and there is no other vector which is a feasible solution of (LP) and has a smaller objective function value, if the problem is a minimization problem (respectively has a bigger objective function value, if the problem is a maximization problem).
- It may be a feasible but non-optimal solution of (LP): the vector is a feasible solution of (LP), but there is another feasible solution with a strictly better objective function value.

The same holds true for integer programs if we replace $\mathbb{R}$ by $\mathbb{Z}$.

2.5 Definition

Suppose we have a linear program (LP).

Exactly one of following properties is true:

- The program is **infeasible**: there is no feasible solution (and thus no feasible optimal solution) for the program.
- The program is **unbounded**: for each value $z \in \mathbb{R}$, there is a feasible solution x with a better objective function value $c^T x < z$ if the program is a minimization problem (respectively $c^T x > z$ if the program is a maximization problem).
- The program is **feasible and not unbounded**: an optimal feasible solution exists.

The same holds true for integer programs if we replace $\mathbb{R}$ by $\mathbb{Z}$.

Note that due to the linearity of the constraints, the value $\sup\{c^T x|$ x is a feasible solution $\}$ will always be taken if the program is neither infeasible nor unbounded for some $x^* \in \mathbb{R}^n$, so the statement is indeed true.

2.6 Definition: Standard Form

We say that a given linear program (LP) is in **standard form** if

- there are no less-or-equal- and greater-or-equal-constraints in (LP),
- all variables are sign-constrained (that means $S = \{1, ..., n\}$ in definition 2.1) and
- (LP) is a minimization problem.

By using the definition of matrix multiplication, we may combine the constraint coefficients a_{ij} to a matrix $A = (a_{ij}) \in \mathbb{R}^{mxn}$ respectively the right-hand-sides of the constraints and objective function coefficients b_j and c_i to vectors $b = (b_1, ..., b_m) \in \mathbb{R}^m$ and $c = (c_1, ..., c_n) \in \mathbb{R}^n$.

Hence, we can write (LP) in the following form:

$$\begin{aligned} &\min\ c^T x \\ &\text{s.t. } Ax = b \\ &\quad\ \ x \geq 0 \end{aligned}$$

We say that a given inter program (IP) is in standard form if the same properties hold true, where $b \in \mathbb{Z}^m$, $A \in \mathbb{Z}^{mxn}$.

2.7 Theorem
For every linear program (LP) (respectively integer program (IP)), there is a linear program (LP') (respectively integer program (IP')) in standard form with an equivalent solution.

Proof & Algorithm:
In the case of a linear program (LP):

- If we maximize the objective function value in (LP), we minimize it in (LP') and use $-c$ instead of c as coefficient vector.

  If x^* was an optimal feasible solution of (LP), there is no x with $c^T x > c^T x^*$. This is equivalent to there is no x with $-c^T x < -c^T x^*$

  If we already minimize in (LP), take c from (LP) without change.

- Take all equal-constraints from (LP) without change.

- For each less-or-equal-constraint $j \in \{i_{max}+1, .., j_{max}\}$, add a surplus variable x_{n+j} with objective function coefficient 0 to (LP') and replace the original constraint
  $a_{j1} * x_{j1} + ... + a_{jn} * x_{jn} \leq b_j$ by
  $a_{j1} * x_{j1} + ... + a_{jn} * x_{jn} + x_{n+j} = b_j$

  Add a sign-constraint for the new surplus variable: $x_{n+j} \geq 0$.

  Since they have coefficient 0, the value of x_{n+j} in an optimal solution of (LP') will be exactly $b_j - A_{.j}x^*$ (which is non-negative since we had a less-or-equal-constraint) where x^* is an optimal solution of (LP).

- For each greater-or-equal-constraint $k \in \{j_{max}+1, .., m\}$, add a slack variable x_{n+k} with objective function coefficient 0 to (LP') and replace the original constraint
  $a_{k1} * x_{k1} + ... + a_{kn} * x_{kn} \geq b_k$ by
  $a_{k1} * x_{k1} + ... + a_{kn} * x_{kn} - x_{n+k} = b_k$

  Add a sign-constraint for the new slack variable: $x_{n+k} \geq 0$.

  With the same argumentation as above, these variables won't influence optimality.

- Take all sign-constraints from (LP) without change.

- Every variable x_i which is not sign-constrained in (LP) (that means $i \notin S$) is replaced by the variables x_i^+ and x_i^-. Every occurrence of x_i in the objective function and constraints is replace by $x_i^+ - x_i^-$.

  The new variables should both by sign-constrained: $x_i^+ \geq 0$, $x_i^- \geq 0$

(LP') is feasible (respectively unbounded) if and only if (LP) was feasible (respectively unbounded).

To get an feasible solution x of (LP) given a feasible solution y of (LP'), set $x_i = y_i$ for $i \in S$ and $x_i = y_i^+ - y_i^-$ for $i \notin S$. If y was an optimal feasible solution, x will be an optimal feasible solution. The components of y corresponding to surplus or slack variables can be omitted.

If we replace (LP) by (IP) and (LP') by (IP'), we get a proof for the integer case, since the coefficients of the new variables in the constraints are integer. □

2.8 Remark

Note that if (LP) is in standard form with $c \geq 0$ component-wise, (LP) cannot be unbounded since no feasible solution can produce a smaller objective function value than 0. Therefore, (LP) has an optimal solution if and only if it has a feasible solution at all.

2.9 Example

The integer program from example 2.3 is equivalent to the integer program
(IP) = min cx s.t. $Ax = b, x \geq 0$ in standard form with

$$c = (100, 80, 0, 0)^T,$$

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix},$$

$$b = (40, 40, 50)^T$$

we get by introducing slack variables x_3, x_4 to transform the less-or-equal-constraints to equal-constraints.

Note that the initial solution v' given in example 2.3 corresponds to the feasible solution $v = (40, 10, 0, 30)^T$ of (IP).

In the rest of this thesis, we will assume that (IP) is a feasible and not unbounded integer program given in standard form.

In the following remarks, I will cite some results on the complexity of optimization problems. This will show that there is currently no known good algorithm for solving integer programs (in contrast to linear optimization, which can be solved efficiently).

2.10 Remark

Although the Simplex algorithm is commonly used for solving linear programs in practice, it may take up to $\binom{n}{m}$ iterations to solve a linear program in standard form with n variables and m constraints, since in the worst case, it iterates through all basis solutions (vectors such that only m out of n components are nonzero). For an example, see [KM72, pp. 159-175].

Karmarkar's algorithm can solve linear programs in a certain form in polynomial time, see [Kar84, p. 373-395] and has thus a better worst-case-complexity, but since the Simplex algorithm is better in the average case, it is not commonly used in practice.

2.11 Example

The following example shows, that rounding an optimal feasible solution of a linear program with integer coefficients does not necessarily give an optimal or even feasible solution of the corresponding integer program.

Consider the linear program (LP) in standard form given by the objective function coefficient vector

$$c = (-1, -3, 0, 0)^T,$$

the constraint matrix

$$A = \begin{pmatrix} 1 & 6 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix},$$

and the right-hand side vector

$$b = (12, 2)^T.$$

Using methods from optimization, we can check, that $x^{(LP)} = (2, \frac{5}{3}, 0, 0)^T$ is an optimal feasible solution with objective function value $c^T x^{(LP)} = -7$.

If we consider the integer program (IP) given by the same matrix and vectors, we can easily see that that the vector $x^{up} = (2, 2, 0, 0)^T$ we get by rounding $x^{(LP)}$ componentwise is not feasible since $1 * 2 + 6 * 2 + 1 * 0 + 0 * 0 = 14 \neq 12$.

Note that the third and fourth variable of the (IP) are surplus variables as in the proof of 2.7. The vector $x^{down} = (2, 1, 4, 0)^T$ we get by rounding the second component of $x^{(LP)}$ down and assigning a suitable value to the first surplus variable is a feasible solution with objective function value $c^T x^{down} = -5$.

The feasible solution $x^{(IP)} = (0, 2, 0, 2)^T$ has objective function value $c^T x^{(IP)} = -6$, so x^{down} is not optimal. (One could check that $x^{(IP)}$ is in fact an optimal solution.)

2.12 Remark

Solving integer programs is a so-called NP-complete problem. (For further information and a proof, see [Kar72, p. 95ff]). That is: if we had a computer, that could "guess" the optimal solution (a non-deterministic Turing machine), we could verify its optimality in polynomial time, but there is currently no known algorithm for solving any NP-complete problem on a real computer (a deterministic Turing machine) in polynomial time.

3 Monomial Orderings, Reduction and Gröbner Bases

Before we can actually try to translate our problem into the language of multivariate polynomials, we will have to introduce several notions and algorithms from computer algebra, which we will need later for solving integer programs.

I will omit some proofs on details in this section, they can be found in any book or lecture notes on computer algebra (for example [Bö]).

3.1 Notation

By

$$K[\underline{t}] = K[t_1, ..., t_n] = \Big\{ \sum_{\alpha \in \mathbb{N}^n} c_\alpha * \underline{t}^\alpha | \text{ only finitely many } c_\alpha \neq 0 \Big\}$$

we define the polynomial ring over a field K in $n > 0$ independent variables.

If I is an ideal generated by $f_1, ..., f_n$ in $K[\underline{t}]$, we write $\langle f_1, ..., f_n \rangle = I \subseteq K[\underline{t}]$.

We call a vector $w \in \mathbb{R}^n$ a weight vector and define for any $f = \sum_\alpha c_\alpha \underline{t}^\alpha$, $f \neq 0$, $f \in K[\underline{t}]$

$$\deg_w(f) = max\{w^T\alpha | c_\alpha \neq 0\}$$

the weighted degree with respect to w.

If we write just deg, we mean the weighted degree with respect to the weight vector $w = (1, 1, ..., 1)$.

If we write $\deg_{t_i}$, we mean the weighted degree with respect to the i^{th} unit vector e_i.

In the following, we will always suppose, that K is any field of characteristic 0.

If we try to generalize algorithms (like polynomial division with remainder) for the polynomial ring in one variable to multivariate polynomials, we notice that these use the notation of leading coefficient and leading monomial which are naturally defined for $K[t]$ but not for $K[\underline{t}]$ if $n > 1$.

Therefore, we define:

3.2 Definition: Monomial Ordering

$>$ is called a **monomial ordering** on $K[\underline{t}]$ if $>$ is a homogeneous relation on the monomials of $K[\underline{t}]$, which fulfills the following properties:

- For monomials $\underline{t}^\alpha$ and $\underline{t}^\beta$ exactly one of the following statements is true:
  $\underline{t}^\alpha > \underline{t}^\beta$, $\underline{t}^\alpha < \underline{t}^\beta$, $\underline{t}^\alpha = \underline{t}^\beta$ (In particular: $>$ is total.)

- $>$ is transitive: If $\underline{t}^\alpha > \underline{t}^\beta$ and $\underline{t}^\beta > \underline{t}^\gamma$, then $\underline{t}^\alpha > \underline{t}^\gamma$
- $>$ respects multiplication:
  If $\underline{t}^\alpha > \underline{t}^\beta$, then for any monomial $\underline{t}^\gamma$ it holds: $\underline{t}^\alpha * \underline{t}^\gamma = \underline{t}^{\alpha+\gamma} > \underline{t}^{\beta+\gamma} = \underline{t}^\beta * \underline{t}^\gamma$

Given a monomial ordering $>$, we can define the usual notions as follows.
Given any $f = \sum_\alpha c_\alpha \underline{t}^\alpha$, $f \neq 0$, $f \in K[\underline{t}]$, we set:

$$\mathrm{LM}_>(f) = max_>\{\underline{t}^\alpha | c_\alpha \neq 0\}$$

the leading monomial,

$$\mathrm{LC}_>(f) = c_\alpha \text{ such that } \underline{t}^\alpha = \mathrm{LM}(f)$$

the leading coefficient and

$$\mathrm{LT}_>(f) = \mathrm{LC}(f) * \mathrm{LT}(f)$$

the leading term of f.

If $>$ is clearly defined by the context, we omit the subscript.

3.3 Definition & Proposition
We say that a monomial ordering $>$ is **global** if it fulfills any of the following equivalent properties:

a) All variables are bigger than the constant monomial, i.e.
   for all $i \in \{1, ..., n\}$: $t_i > 1 = t^0$.

b) All non-constant monomials are bigger than the constant monomial, i.e.
   for all $\alpha \in \mathbb{N}^n, \alpha \neq 0$: $\underline{t}^\alpha > 1$.

c) $>$ is compatible with divisibility, i.e.
   if $\underline{t}^\alpha | \underline{t}^\beta$ (that is $\alpha \geq \beta$ component-wise) and $\alpha \neq \beta$, then $\underline{t}^\alpha > \underline{t}^\beta$.

d) $>$ is a well-ordering, i.e.
   every non-empty set of monomials has a smallest element.

Proof of equivalency:

a) => b)

Induction on k = deg($\underline{t}^\alpha$).
<u>k = 1:</u> By assumption.
<u>k > 1:</u> Choose i such that $\alpha_i \neq 0$. We have $\underline{t}^{\alpha - e_i} > 1$ by induction, thus $\underline{t}^\alpha > t^{e_i}$ by compatibility with multiplication. Since $\underline{t}^{e_i} > 1$, $\underline{t}^\alpha > 1$ follows by transitivity.

b) => c)

Suppose $\alpha \geq \beta$ component-wise and $\alpha \neq \beta$, set $\gamma = \alpha - \beta \neq 0$. We have $\underline{t}^\gamma > 1$ by assumption, so by compatibility with multiplication, $\underline{t}^\alpha = \underline{t}^{\gamma+\beta} > \underline{t}^\beta$ follows.

c) => d)

We can prove this by showing that there are only finitely many elements which are minimal with respect to divisibility in every non-empty set of monomials. For a detailed proof, see [Bö, p. 40f].

d) => a)

Set $S = \{t_i^j | j \in \mathbb{N}\}$ for any $i \in \{1, ..., n\}$. By assumption, this set has a minimal element t_i^k. Suppose this element would not be $1 = t_i^0$. Then we have $t_i^k < 1$ (since 1 divides any t_i and is not the smallest element) and thus by compatibility with multiplication $t_i^{2k} < t_i^k$. This is a contradiction to the choice of t_i^k as smallest element. So for every variable, $t_i^0 = 1$ is the smallest element in the set.

□

3.4 Example

The **degree reverse lexicographical ordering** $>_{dp}$ defined by

$\underline{t}^\alpha >_{dp} \underline{t}^\beta :\Leftrightarrow \deg(t^\alpha) > \deg(\underline{t}^\beta)$

or (degrees equal and rightmost non-zero entry of $\alpha - \beta$ is negative)

is a global monomial ordering.

3.5 Example

Suppose $t_{i_1} < ... < t_{i_n} < 1$ is an ordering of the variables of $K[\underline{t}]$ with $\{i_1, ..., i_n\} = \{1, ..., n\}$. Then the **local reverse lexicographic ordering** $>_{ls}$ with respect to this ordering defined by

$\underline{t}^\alpha >_{ls} \underline{t}^\beta :\Leftrightarrow \deg_{t_{i_k}}(t^\alpha) = \deg_{t_{i_k}}(t^\beta)$ for $k = 1, ..., r-1$ and $\deg_{t_{i_r}}(t^\alpha) < \deg_{t_{i_r}}(t^\beta)$

is a local monomial ordering (that is: $1 >_{ls} t_i$ for all $i \in \{1, ..., n\}$, in particular $>_{ls}$ is not global).

3.6 Remark

Note that the monomials of $K[\underline{t}]$ (with the multiplication of polynomials) are isomorphic to $\mathbb{N}^n$ (with component-wise addition) as semi-groups via the map $\underline{t}^\alpha \mapsto \alpha$.

Therefore, any monomial ordering $>$ induces an ordering of vectors in $\mathbb{N}^n$ with the same properties (if we replace the multiplication of monomials by the addition of vectors), which we will also call $>$.

3.7 Example

Note that any vector $w \in \mathbb{R}^n$ gives a partial ordering on the monomials of $K[\underline{t}]$ by setting $\underline{t}^\alpha >_w \underline{t}^\beta :\Leftrightarrow w^T\alpha >_{\mathbb{R}} w^T\beta$.

If we choose any monomial ordering $>$ as tie-break, we get an induced monomial ordering $>_w$:

$$\underline{t}^\alpha >_w t^\beta :\Leftrightarrow (w^T\alpha >_{\mathbb{R}} w^T\beta) \text{ or } (w^T\alpha = w^T\beta \text{ and } \underline{t}^\alpha > \underline{t}^\beta)$$

(The first two properties of a monomial ordering are induced by the orderings $>_{\mathbb{R}}$ and $>$. The compatibility with multiplication is given since $>$ is compatible and if we multiply both sides with some $\underline{t}^\gamma$, their value increases by the constant $w^T\gamma$.)

Note that if $w > 0$ component-wise, $>_w$ is global, even if we choose a non-global tie-break (like $>_{ls}$): $w^T 0 = 0$ and for any non-constant monomial $\underline{t}^\alpha \neq 1$, $w^T\alpha$ has a positve value.

If $w \geq 0$ component-wise, $>_w$ is global if for all i with $w_i = 0$, $x_i > 0$ holds (where $>$ is the chosen tie-break). This is for example the case if the tie-break is a global ordering (like $>_{dp}$).

3.8 Definition
Let (IP) = $\min cx$ s.t. $Ax = b, x \geq 0$ be an integer program. We say that a monomial ordering $>$ is compatible to (IP) if $\underline{t}^v > \underline{t}^u$ for all $v, u \in \mathbb{N}^n$ with $c^T v > c^T u$ and $Av = Au = b$.

3.9 Example
Let (IP) = $\min cx$ s.t. $Ax = b, x \geq 0$ be an integer program, then any monomial ordering $>_c$ induced by c with an arbitrary monomial ordering as tie-break (as in example 3.7) is compatible with (IP), since $\underline{t}^v >_c \underline{t}^u$ holds for all $v, u \in \mathbb{N}^n$ with $c^T v > c^T u$.

Note that this monomial ordering may not be global, as shown above.

With this knowledge, we can take the first step in translating our optimization problem into an algebraic one:

3.10 Theorem
Let (IP) be an integer program in standard form and $>$ a monomial ordering compatible to (IP).

If the monomial $\underline{t}^x \in K[\underline{t}]$ is minimal with respect to $>$ among those $\underline{t}^y$ with $Ay = b$, then x is an optimal feasible solution for (IP).

Proof:
Since (IP) is in standard form, all variables are sign-constrained. Therefore, we loose

nothing by the restriction to monomials with exponent vectors $y \geq 0$ component-wise.

Suppose there is a feasible solution y with $c^T y < c^T x$, then $\underline{t}^y < \underline{t}^x$ by the compatibility of the monomial ordering, so $\underline{t}^x$ was not minimal with respect to $>$. □

Note that there may be several optimal feasible solutions with the same objective function value, but our monomial ordering will favor one of the corresponding monomials. Since it is sufficient to find one arbitrary optimal feasible solution, the following theorem makes sense.

3.11 Corollary

With the same setup as in theorem 3.10, we can reformulate our optimization problem (IP) as

$$\min_{x \in \mathbb{N}^n} \{\underline{t}^x | Ax = b\}.$$

Using the notion of a monomial ordering, we can generalize polynomial division to multivariate polynomials.

3.12 Algorithm: Reduced Buchberger Normal Form

Input:

$f \in K[\underline{t}]$,
G finite set of non-zero polynomials in $K[\underline{t}]$,
$>$ global monomial ordering.

Output:

$\mathrm{redNF}_>(f, G)$, the reduced Buchberger normal form of f with respect to G and $>$.

$r := 0$
while $f \neq 0$ **do**
  if $\mathrm{LM}(f)$ *divisible by* $\mathrm{LM}(g)$ *for some* $g \in G$ **then**
    $f := f - \frac{\mathrm{LT}(f)}{\mathrm{LT}(g)} * g$
  else
    $r := r + \mathrm{LT}(f)$
    $f := f - \mathrm{LT}(f)$
  end
end
return r

Proof of termination:

$\mathrm{LM}(f)$ gets smaller in every step with respect to $>$, so the algorithm terminates after finitely many steps, since $>$ is global (see definition 3.3). □

3.13 Definition: Leading Ideal

For $G \subseteq K[\underline{t}]$ and a monomial ordering $>$ define

$$L(G) = L_>(G) = \langle \mathrm{LM}(f) \mid f \in G \rangle$$

the **leading ideal** of G with respect to $>$.

3.14 Proposition

For f, G and $>$ as in the input of algorithm 3.12, $r = \mathrm{redNF}_>(f, G)$ fulfills the following properties:

a) If $r \neq 0$, then $\mathrm{LM}(r) \notin L(G)$.

b) r is tail-reduced: no term in $\mathrm{tail}(r) = r - \mathrm{LT}(r)$ is in $L(G)$.

Proof:

a)
Suppose $r \neq 0$, $\mathrm{LM}(r) \in L(G)$. Since membership can be decided by division in ideals generated by monomials and $L(G)$ is generated by the leading monomials of the $g \in G$, $\mathrm{LM}(r)$ would be divisible by $\mathrm{LM}(g)$ for some $g \in G$.
This is a contradiction, because in this situation, the term $\mathrm{LM}(r)$ of f would not be put in the remainder, but canceled out in the algorithm.

b)
If some term would be in $L(G)$, it would be divisible by some $g \in G$. Then the term of f would not by put into the remainder, but canceled out in the algorithm.

□

3.15 Example

For $f = t_1$ and $G = \{t_1 t_2 + t_1, t_2\}$, we get (using $>_{dp}$ from example 3.4):
$\mathrm{redNF}(f, G) = t_1$.
Note that $f = 1 * (t_1 t_2 + t_1) + (-t_1) * (t_2)$, so we get a non-zero remainder altough $f \in \langle G \rangle$. This occurs since $t_1 \in L(\langle G \rangle)$ but $t_1 \notin L(G)$.

3.16 Example

Consider the global monomial ordering $>_c$ induced by the vector $c = (100, 80, 0, 0)^T$ with $>_{dp}$ as tie-break.
Let $f = \underline{t}^v = t_1^{40} t_2^{10} t_4^{30}$ with $v = (40, 10, 0, 30)^T$ from example 2.9 and
$G = \{t_1 t_4 - t_2 t_3\}$.
We get $\mathrm{redNF}(f, G) = t_1^{10} t_2^{40} t_3^{30}$.

3.17 Definition: Gröbner Basis

Let $I \subseteq K[\underline{t}]$ be an ideal and $>$ a global monomial ordering.
We call a finite set $G \subseteq I$ **Gröbner basis** of I (with respect to $>$) if $L(G) = L(I)$.

We call a Gröbner basis G minimal if there are no two elements $f \neq g$ in G such that $LT(f)$ divides $LT(g)$.

We call a minimal Gröbner basis G reduced, if for all $g \in G$:

- $LC(g) = 1$ and
- no term of tail(g) $= g - LT(g)$ is in $L(G)$.

3.18 Theorem

Let $I \subset K[\underline{t}]$ be an ideal and $>$ a global monomial ordering. Then there exists a reduced Gröbner basis for I with respect to $>$.

Proof:
$L(I)$ is a monomial ideal in $K[\underline{t}]$ and thus has finitely many monomial generators since $K[\underline{t}]$ is noetherian (by Hilbert's Basis Theorem and results on monomial ideals, see [Bö, p. 31f resp. p. 41]). By definition of $L(I)$, there are polynomials $g \in I$, such that these generators are their leading monomials. These g form a Gröbner basis G of I.

If the leading monomial of $g \in G$ divides the leading monomial of some $f \in G$, we have $\langle \text{LM}(f) \rangle \subseteq \langle \text{LM}(g) \rangle$, so we can remove f from G without changing the leading ideal. If we remove all such elements, we get a minimal Gröbner basis.

We normalize every g by dividing by LC(g). If we replace every g by redNF($g, G \backslash \{g\}$), one gets a reduced Gröbner basis, since the reduced Buchberger normal form is tail-reduced with respect to $G \backslash \{g\}$ (and that no term of tail(g) is divisible by $LM(g)$ is clear since $>$ is a global monomial ordering and thus compatible with divisibility). □

3.19 Remark

One can show that the reduced Gröbner basis of some ideal $I \subseteq K[\underline{t}]$ is uniquely determined by I and $>$, see [Bö, p. 55]. Therefore, it is correct to speak of "the" reduced Gröbner basis of I with respect to $>$.

3.20 Remark

Note that a Gröbner basis G of an ideal $I \subseteq K[\underline{t}]$ is always a set of generators, that is $\langle G \rangle = I$, as proven in [Bö, p. 49]

3.21 Definition

Given monomials $\underline{t}^\alpha, \underline{t}^\beta \in K[\underline{t}]$, we may define $\gamma \in \mathbb{N}^n$ by $\gamma_i = \max\{\alpha_i, \beta_i\}$ and call $\operatorname{lcm}(\underline{t}^\alpha, \underline{t}^\beta) = \underline{t}^\gamma$ the **least common multiple** of $\underline{t}^\alpha$ and $\underline{t}^\beta$.

We call $\underline{t}^\alpha$ and $\underline{t}^\beta$ **coprime** if $\operatorname{lcm}(\underline{t}^\alpha, \underline{t}^\beta) = \underline{t}^\alpha * \underline{t}^\beta$

3.22 Algorithm: Buchberger

Input:

$I = \langle g_1, \ldots, g_k \rangle \subseteq K[\underline{t}]$ ideal,

$>$ global monomial ordering.

Output:

G Gröbner basis of I with respect to $>$.

$G = \{g_1, \ldots, g_k\}$
repeat
  $H := G$
  forall $f, g \in H$ **do**
    $\operatorname{spoly}(f, g) := \frac{\operatorname{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LT}(f)} f - \frac{\operatorname{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LT}(g)} g$
    $r := \operatorname{redNF}(\operatorname{spoly}(f, g), H)$
    if $r \neq 0$ **then**
      $G := G \cup \{r\}$
    end
  end
until $G = H$;
return G

Proof of termination:

Suppose we add some $r \neq 0$ to G during the algorithm. By the properties of the normal form from proposition 3.14, we have $\mathrm{LM}(r) \notin L(H)$, hence $L(H) \subsetneqq L(H \cup \{r\})$.

That means every time we add an element to G, we increase its leading ideal. This gives an ascending chain of ideals, which has to stop since $K[\underline{t}]$ is noetherian. Therefore, the algorithm terminates after finitely many steps. □

Proof of correctness:

One may show, that $\langle G \rangle = I$ and $\operatorname{redNF}(\operatorname{spoly}(f, g), G) = 0$ for all $f, g \in G$ is an equivalent condition for being a Gröbner basis. For a proof, see [Bö, p. 148f]. □

3.23 Example

Suppose $I = \langle f \rangle \subseteq K[\underline{t}]$ is a principal ideal. Then $G = \left\{ \frac{f}{\mathrm{LC}(f)} \right\}$ is a reduced Gröbner basis of I with respect to any global monomial ordering by definition.

4 A generic Algorithm for solving Integer Programs

In this section, we will associate an ideal to the constraint matrix and then show that this ideal (respectively its reduced Gröbner basis) has nice properties, which allow us to reduce any feasible solution of an integer program to an optimal one.

Since this ideal is spanned by binomials, we will at first look at such polynomials.

4.1 Definition

Suppose $x \in \mathbb{Z}^n$, then we define x^+ and $x^- \in \mathbb{N}^n$ by

$$x_i^+ = \begin{cases} 0 & , x_i \leq 0, \\ x_i & , x_i \geq 0, \end{cases}$$

respectively

$$x_i^- = \begin{cases} -x_i & , x_i \leq 0, \\ 0 & , x_i \geq 0. \end{cases}$$

4.2 Definition: Pure and Primitive Binomials

Suppose $f \in K[\underline{t}]$. We call f a **binomial** if it is of the form $f = a\underline{t}^u + b\underline{t}^v$ for some $a, b \in K$, $u, v \in \mathbb{N}^n$.

We call f a **pure binomial** if f is a binomial and $a = 1, b = -1$, that is: $f = \underline{t}^u - \underline{t}^v$.

We call f a **primitive binomial** if f is a pure binomial and $\underline{t}^u$ and $\underline{t}^v$ are coprime.

As the following lemma shows, definition 4.2 and 4.1 define indeed the same thing.

4.3 Lemma

a) For any $x \in \mathbb{Z}^n$, x^+ and x^- are the unique vectors in $\mathbb{N}^n$ with $x = x^+ - x^-$ and $x_i^+ * x_i^- = 0$ for all $i \in \{1, ..., n\}$.

b) $f \in K[\underline{t}]$ is a primitve binomial if and only if there is an $x \in \mathbb{Z}^n$ with $f = \underline{t}^{x^+} - \underline{t}^{x^-}$.

Proof:

a)

By definition, it is clear that x^+ and x^- have the desired properties.
Suppose u, v would have these properties, too. Then for each component, $x_i = u_i - v_i$ and at least one of them is zero. It follows that $u_i = x_i = x_i^+$ if $x_i \geq 0$ and $v_i = -x_i = x_i^-$ if $x_i \leq 0$ and the other one is zero.

b) "=>"

Suppose $f = \underline{t}^u - \underline{t}^v$ is a primitive binomial. Set $x = u - v$. Since $\underline{t}^u$ and $\underline{t}^v$ are coprime, $u_i * v_i = 0$ for all $i \in \{1, ..., n\}$. Using a), we get that $u = x^+$ and $v = x^-$.

"<="

Suppose that $f = \underline{t}^{x^+} - \underline{t}^{x^-}$. Since $x_i^+ * x_i^- = 0$ for all $i \in \{1, ..., n\}$, $\underline{t}^{x^+}$ and $\underline{t}^{x^-}$ are coprime. Thus, f is a primitive binomial.

□

The following proposition will show, that at least pure binomials behave well with respect to the algorithms introduced in the third section.

4.4 Proposition

Let $f \in K[\underline{t}]$, $G \subset K[\underline{t}]$ a finite set, $>$ a global monomial ordering.

a) If $f = \underline{t}^x$ is a monomial and G is a finite set of pure binomials, then $\text{redNF}(f, G)$ is a monomial.

b) If $f = \underline{t}^u - \underline{t}^v$ is a pure binomial and G is a finite set of pure binomials, then $\text{redNF}(f, G)$ is a pure binomial.

c) If G is a finite set of pure binomials, then $\langle G \rangle$ has a (minimal / reduced) Gröbner basis consisting of pure binomials.

Proof:

a)

Suppose $f = \underline{t}^x$, $g = \underline{t}^u - \underline{t}^v$ and we are in the if-case of algorithm 3.12. Suppose $\text{LT}(g) = \underline{t}^u$ divides f. Then the next step will set

new $f = f - \frac{\text{LT}(f)}{\text{LT}(g)} * g$

$= \underline{t}^x - \frac{\underline{t}^x}{\underline{t}^u}(\underline{t}^u - \underline{t}^v)$

$= \underline{t}^x - \underline{t}^{x-u} * (\underline{t}^u - \underline{t}^v)$

$= \underline{t}^x - \underline{t}^{x-u+u} + \underline{t}^{x-u+v}$

$= \underline{t}^{x-u+v}$.

With the same argument, we get for $\text{LT}(g) = -\underline{t}^v$ dividing f, that new $f = \underline{t}^{x-v+u}$. Thus, we replace the monomial f in each if-step by another monomial. As soon as the else-step occurs, the algorithm will terminate with a monomial as output since $\text{LT}(f) = f$.

b)

Suppose $f = \underline{t}^x - \underline{t}^y$, $g = \underline{t}^u - \underline{t}^v$ and we are in the if-case of algorithm 3.12.

Case $\text{LT}(f) = \underline{t}^x, \text{LT}(g) = \underline{t}^u$:

new $f = f - \frac{\text{LT}(f)}{\text{LT}(g)} * g$

$= \underline{t}^x - \underline{t}^y - \frac{\underline{t}^x}{\underline{t}^u}(\underline{t}^u - \underline{t}^v)$

$= \underline{t}^x - \underline{t}^y - \underline{t}^{x-u} * (\underline{t}^u - \underline{t}^v)$

$= \underline{t}^{x} - \underline{t}^{y} - \underline{t}^{x-u+u} + \underline{t}^{x-u+v}$
$= \underline{t}^{x-u+v} - \underline{t}^{y}$.
Analogously:
Case $\mathrm{LT}(f) = \underline{t}^{x}, \mathrm{LT}(g) = -\underline{t}^{v}$: new $f = \underline{t}^{x-v+u} - \underline{t}^{y}$.
Case $\mathrm{LT}(f) = -\underline{t}^{y}, \mathrm{LT}(g) = \underline{t}^{u}$: new $f = \underline{t}^{x} - \underline{t}^{y-u+v}$.
Case $\mathrm{LT}(f) = -\underline{t}^{y}, \mathrm{LT}(g) = -\underline{t}^{v}$: new $f = \underline{t}^{x} - \underline{t}^{y+u-v}$.
So as long as we stay in the if-step, we will always get pure binomials. When we enter the else-step for the first time (after having subtracted some monomial), we continue with a monomial which means we are in the case of a). (By replacing $\underline{t}^{x}$ by $-\underline{t}^{x}$ in the proof of a), we see that even if we start with a signed monomial, the algorithm preserves this sign). By adding these two monomials up, we see that $\mathrm{redNF}(f, G)$ is a pure binomial.

c)

Suppose $f = \underline{t}^{x} - \underline{t}^{y}$, $g = \underline{t}^{u} - \underline{t}^{v}$. We compute $\mathrm{spoly}(f, g)$ as in algorithm 3.22.
In the case $\mathrm{LT}(f) = \underline{t}^{x}, \mathrm{LT}(g) = \underline{t}^{u}$, we get
$\mathrm{spoly}(f, g) = \frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LT}(f)} f - \frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LT}(g)} g$
$= \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{x}} (\underline{t}^{x} - \underline{t}^{y}) - \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{u}} (\underline{t}^{u} - \underline{t}^{v})$
$= \mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u}) - \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{x}} \underline{t}^{y} - \mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u}) + \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{u}} \underline{t}^{v}$
$= \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{u}} \underline{t}^{v} - \frac{\mathrm{lcm}(\underline{t}^{x}, \underline{t}^{u})}{\underline{t}^{x}} \underline{t}^{y}$
which is a pure binomial.
Analogously, we see that we get a pure binomial in the other cases too, since the parts corresponding to the lead terms of each binomial cancel out.
With the statement from b), we get that if we start with a set of pure binomials, we get a Gröbner basis consisting of pure binomials by applying algorithm 3.22.

If we omit elements to minimize the Gröbner basis as in the proof of thereom 3.18, this stays true, so we can also get a minimal Gröbner basis consisting of such binomials.

By normalizing, we may switch the signs of the terms, but the binomials stay in the required form. We already know from b) that applying the reduction-algorithm will give us pure binomials, so we also have a reduced Gröbner basis consisting of such binomials.

□

4.5 Example
For $f = t_1^3 - t_2^2$ and $G = \{t_1^2 - t_2\}$, we get (using $>_{dp}$ from example 3.4):
$\mathrm{redNF}(f, G) = t_1 t_2 - t_2^2$.

So even if we start with primitive binomials, the algorithms may not preserve the co-primeness. We will get pure binomials, but there are cases in which they are not primitive.

We will see, that our further progress on the way to represent the optimization problem as an algebraic one will give us an example for an ideal generated by primitive binomials where the reduced Gröbner basis consists of primitive binomials, too.

4.6 Definition: Lattice

A $\mathbb{Z}$-Module is called **lattice** if it is free and finitely generated (that means, it has a finite basis).
We call such a finite basis a **lattice basis** for the lattice.

4.7 Example

Suppose $A \in \mathbb{Z}^{m \times n}$ is a matrix with integer entries. Then

$$\mathrm{Ker}(A) = \{x \in \mathbb{Z}^n | Ax = 0\} \subseteq \mathbb{Z}^n$$

is a $\mathbb{Z}$-submodule of $\mathbb{Z}^n$ (Proof as for $\mathrm{Ker}(A) \subseteq K^n$ subvectorspace).
Since $\mathbb{Z}^n$ is noetherian (see [Mar, p. 59]), $\mathrm{Ker}(A)$ is finitely generated. As a finitely generated torsion-free module over the principal ideal domain $\mathbb{Z}$, it is free (see [Bö, p. 103]). Therefore, $\mathrm{Ker}(A)$ is a lattice.

4.8 Remark

Every lattice has a finite basis by definition, but on our way to compute an optimal solution for integer optimization problems, we will need a way to compute it.
This task can be done in polynomial time using the LLL algorithm, see [LLL82, p. 515-534].

4.9 Definition: Toric Ideal associated to A

Suppose $A \in \mathbb{Z}^{m \times n}$ is a matrix with integer entries. Then we define the ideal I_A by

$$I_A = \langle \underline{t}^{x^+} - \underline{t}^{x^-} | x \in \mathrm{Ker}(A) \rangle \subseteq K[\underline{t}]$$

and call it the **toric ideal associated to** A.

As we can see, the ideal is generated by (in general) infinitely many primitive binomials. Note that it is not immediately clear how to compute I_A, but we are able to compute

$$I_{A_0} = \langle \underline{t}^{x^+} - \underline{t}^{x^-} | x \in U \rangle \subseteq K[\underline{t}]$$

where $U \subset \mathbb{Z}^n$ is a lattice basis for $\mathrm{Ker}(A)$ (which we may compute using the LLL algorithm from the remark above).
Since those ideals may differ and we really need I_A for the final algorithm, we will study later how to compute I_A from I_{A_0}.

4.10 Example

Let $A = \mathbb{1}_n \in \mathbb{Z}^{nxn}$ be the unit matrix of rank n.
Obviously, $\mathrm{Ker}(A) = 0$, so $I_A = \langle \underline{t}^0 - \underline{t}^0 \rangle = \langle 0 \rangle$.

4.11 Example

One can check using the LLL algorithm that the kernel of the constraint matrix A of the integer program in standard form from example 2.9 is spanned by the vector

$$x = (1, -1, -1, 1)^T.$$

Therefor, we can compute at least

$$I_{A_0} = \langle \underline{t}^{x^+} - \underline{t}^{x^-} \rangle = \langle \underline{t}^{(1,0,0,1)^T} - \underline{t}^{(0,1,1,0)^T} \rangle = \langle t_1 t_4 - t_2 t_3 \rangle.$$

As the following lemma shows, I_A arises in a natural way if we want to translate the kernel of an integer matrix into algebraic terms.

4.12 Lemma

Let $A \in \mathbb{Z}^{mxn}$, $x \in \mathbb{Z}^n$

a) Define

$$\phi : K[\underline{t}] \to K[s_1, s_1^{-1}, \ldots, s_m, s_m^{-1}]$$

$$t_i \mapsto s^{a_i}$$

as the K-algebra homomorphism which sends the i^{th} Variable t_i to s^{a_i}, where a_i is the i^{th} column of A.
Then: $I_A = \mathrm{Ker}(\phi)$

b) $x \in \mathrm{Ker}(A) \Leftrightarrow \underline{t}^{x^+} - \underline{t}^{x^-} \in I_A$

Proof:

a) "$\subseteq$"

Let $x \in \mathrm{Ker}(A)$, then
$0 = \sum\limits_{j=1}^{m} x_j a_j = \sum\limits_{j=1}^{m} x_j^+ a_j - \sum\limits_{j=1}^{m} x_j^- a_j$, so $\sum\limits_{j=1}^{m} x_j^+ a_j = \sum\limits_{j=1}^{m} x_j^- a_j$.

Thus: $\phi(\underline{t}^{x^+} - \underline{t}^{x^-}) = \underline{s}^{\sum_{j=1}^{m} x_j^+ a_j} - \underline{s}^{\sum_{j=1}^{m} x_j^- a_j} = 0$.
Since I_A is generated by all $\underline{t}^{x^+} - \underline{t}^{x^-}$ where $x \in \mathrm{Ker}(A)$, $I_A \subseteq \mathrm{Ker}(\phi)$ follows.

"$\supseteq$"

Suppose $Ker(\phi)\backslash I_A$ is nonempty. Choose any global monomial ordering $>$ and choose $f \in \mathrm{Ker}(\phi)\backslash I_A$ with $\mathrm{LT}(f) = \mathrm{LM}(f) = \underline{t}^x$ minimal with respect to $>$ for some $x \in \mathbb{Z}^n, x \neq 0$. (This is possible since for any $g \in \mathrm{Ker}(\phi)\backslash I_A$, $\frac{g}{\mathrm{LC}(g)} \in \mathrm{Ker}(\phi)\backslash I_A$, too.)

Since ϕ sends monomials to monomials and K-linear combinations of different monomials are never zero, it follows from $\phi(f) = 0$ that there is another monomial $\underline{t}^v$ in tail(f) with $\phi(\underline{t}^x) = \phi(\underline{t}^v)$.
In particular we have : $f' = f - \underline{t}^x - \underline{t}^v \in \mathrm{Ker}(\phi)$ (since the images of the monomials cancel each other out) and $\underline{t}^x - \underline{t}^v \in \mathrm{Ker}(\phi)$ (since $\mathrm{Ker}(\phi)$ is an ideal).
We know $\phi(\underline{t}^x - \underline{t}^v) = \underline{s}^{\sum_{j=1}^{m} x_j a_j} - \underline{s}^{\sum_{j=1}^{m} v_j a_j} = 0$, thus $\sum_{j=1}^{m} x_j a_j = \sum_{j=1}^{m} v_j a_j$, so $x - v$ is in $\mathrm{Ker}(A)$, $\underline{t}^x - \underline{t}^v \in I_A$ (Compare with "$\subseteq$").
Since $f \notin I_A$ and $\underline{t}^x - \underline{t}^v \in I_A$, their difference $f' \notin I_A$. We have $\mathrm{LM}(f') < \mathrm{LM}(f)$ (since we canceled out the original leading monomial), which is a contradiction to the choice of f.

b) "=>"

Follows directly from the definition of I_A.

"<="

Suppose $\underline{t}^{x^+} - \underline{t}^{x^-}$ is in I_A, then $\phi(\underline{t}^{x^+} - \underline{t}^{x^-}) = \underline{s}^{\sum_{j=1}^{m} x_j^+ a_j} - \underline{s}^{\sum_{j=1}^{m} x_j^- a_j} = 0$, thus $\sum_{j=1}^{m} x_j^+ a_j = \sum_{j=1}^{m} x_j^- a_j$, so $x^+ - x^- = x$ is in $\mathrm{Ker}(A)$.

□

In the following part, we want to show the correspondence between I_A and the solution of the optimization problem. As you may have noted, the definition of I_A does not use the right-hand side b of the equation $Ax = b$, which is defining the space of feasible solutions of an integer program.

Indeed, we can generalize our problem to solving families of linear programs where b is variable.

4.13 Definition: Test Set

Suppose $A \in \mathbb{Z}^{mxn}$, $c \in \mathbb{R}^n_{\geq 0}$. Then

$$IP_{A,c} = \{IP_{A,c}(b) | b \in \mathbb{Z}^m\} = \{\min c^T x \text{ s.t. } Ax = b, x \geq 0 | b \in \mathbb{Z}^m\}$$

is the **family of linear programs** given by A and c.

Let $>_c$ be a global monomial ordering with $\underline{t}^x > \underline{t}^y$ for x, y with $c^T x > c^T y$ (which then also induces an ordering $>_c$ of the vectors in $\mathbb{N}^n$ as in remark 3.6).

We call a finite set $X \subseteq \{x \in \mathbb{Z}^n | Ax = 0, x^+ >_c x^-\}$ a **test set** for $IP_{A,c}$ if it fulfills the following properties:

- Suppose v is a feasible solution for some $IP_{A,c}(b)$, but not the solution such that its corresponding monomial is minimal with respect to $>_c$ among the feasible ones (see theorem 3.10 and corollary 3.11). Then there is an $x \in X$ such that $v - x$ is also a feasible solution of $IP_{A,c}(b)$.
- Suppose v^* is the feasible solution of some $IP_{A,c}(b)$ such that its corresponding monomial is minimal among the feasible ones (in particular: v^* is an optimal feasible solution). Then there is no $x \in X$ such that $v^* - x$ is a feasible solution of $IP_{A,c}(b)$.

As the name suggests, we can indeed check optimality using test sets: for a feasible solution v, there is either no $x \in X$ with $v - x$ feasible (thus: v is optimal), or there is such an x and we can replace v by $v - x$ and iterate.

In the next step, we want to show that the toric ideal I_A associated to the constraint matrix of an integer program gives us a test set. Therefore, we need the following statement, which we will prove later in the next section, where we will show how to compute I_A.

4.14 Assumption
Let $A \in \mathbb{Z}^{mxn}$, $>$ any global monomial ordering. Then the reduced Gröbner basis of I_A with respect to $>$ consists of primitive binomials.

4.15 Theorem
Suppose $A \in \mathbb{Z}^{mxn}$, $c \in \mathbb{R}^n$, $>_c$ a global monomial ordering with $c^T\alpha > c^T\beta \Rightarrow \underline{t}^\alpha >_c \underline{t}^\beta$ (in particular: $c \geq 0$ component-wise, compare with example 3.7). Suppose $G = \{\underline{t}^{x_i^+} - \underline{t}^{x_i^-} | i \in \{1, ..., k\}\}$ is the reduced Gröbner basis of I_A with respect to $>_c$.

Then the exponent vectors $X = \{x_i | i \in \{1, ..., k\}\}$ form a test set for $IP_{A,c}$.

Proof:
Under the assumption 4.14, the reduced Gröbner basis of I_A is indeed in the given form, as we have shown in 4.3.

A Gröbner basis is a finite set by definition, therefore X too.
We have shown in lemma 4.12 that the $x \in X$ are in $\text{Ker}(A)$, so $Ax = 0$ is fulfilled.
We have $\text{LM}(\underline{t}^{x_i^+} - \underline{t}^{x_i^-}) = \underline{t}^{x_i^+}$, since the elements of the reduced Gröbner basis are monic. Hence, $x_i^+ >_c x_i^-$.

- Suppose v is a feasible solution for some $IP_{A,c}(b)$, but not the exponent vector of the solution described in 3.11.
  Suppose v^* is this optimal solution of $IP_{A,c}(b)$.
  By $A(v - v^*) = Av - Av^* = b - b = 0$, we see that $y = (v - v^*) \in \text{Ker}(A)$, so with 4.12, we have $\underline{t}^{y^+} - \underline{t}^{y^-} \in I_A$.
  Since $v^* <_c v$, we have $\text{LT}(\underline{t}^{y^+} - \underline{t}^{y^-}) = \underline{t}^{y^+}$.

  Since $L(G)$ spans the leading ideal of I_A, there is an x in X such that $\text{LT}(\underline{t}^{x^+} - \underline{t}^{x^-}) = \underline{t}^{x^+}$ divides $\text{LT}(\underline{t}^{y^+} - \underline{t}^{y^-}) = \underline{t}^{y^+}$, that is $x^+ \leq y^+$ component-wise.
  We get $x^+ \leq y^+ = (v - v^*)^+ \leq v^+ = v$ since $v, v^* \geq 0$ as feasible solutions, so $v - x = (v - x^+) + x^- \geq 0$.
  Since $Av = b$ and $Ax = 0$ (since $x \in \text{Ker}(A)$), we have $A(v - x) = b$, so $(v - x)$ is a feasible solution.

- Suppose v^* is the optimal solution for some $IP_{A,c}(b)$ described by corollary 3.11. Suppose there is an $x \in X$, such that $(v^* - x)$ is feasible. Since $x^+ >_c x^-$, we have $x = x^+ - x^- >_c 0$ respectively $0 >_c -x$. $v^* >_c v^* - x$ follows, but this is a contradiction to the choice of v^*.

□

4.16 Theorem
In the same setting as in theorem 4.15 and for v any feasible solution for some $IP_{A,c}(b)$, the exponent vector v^* of $\text{redNF}(\underline{t}^v, G) = \underline{t}^{v^*}$ is an optimal solution for $IP_{A,c}(b)$.

Proof:
We have shown in proposition 4.4, that $\text{redNF}(\underline{t}^v, G)$ is indeed a monomial. We saw in the proof, that reducing $\underline{t}^v$ by $\underline{t}^{x^+} - \underline{t}^{x^-}$ (when the leading term is $\underline{t}^{x^+}$, which is true for $x \in X$, since X is a test set) gives $\underline{t}^{v-x^++x^-} = \underline{t}^{v-(x^+-x^-)}$. Since $A(v - (x^+ - x^-)) = A(v - x) = Av - Ax = b - 0 = b$ and we get $\text{redNF}(\underline{t}^v, G)$ by iterating such reductions, the exponent vector of $\text{redNF}(\underline{t}^v, G)$ is a feasible solution.

Suppose the exponent vector v^* of $\text{redNF}(\underline{t}^v, G)$ would not be optimal. Since X is a test set, we would have some $x \in X$ such that $v^* - x$ is feasible. In particular $v^* - x^+ \geq 0$ component-wise (since we assume that the Gröbner basis consists of primitive binomials). It follows that $\mathrm{LT}(\underline{t}^{x^+} - \underline{t}^{x^-}) = \underline{t}^{x^+}$ would divide $\underline{t}^{v^*}$, but then by the properties of redNF from proposition 3.14, $\underline{t}^{v^*}$ could not be the reduced normal form of $\underline{t}^{v}$. □

This proves the correctness of the following algorithm in situations where the assumption 4.14 holds.

4.17 Algorithm: Generic algorithm for solving IPs

Input:

$A \in \mathbb{Z}^{m \times n}$,

$c \in \mathbb{R}^n_{\geq 0}$,

$>_c$ a global monomial ordering with $c\alpha > c\beta \Rightarrow \underline{t}^{\alpha} >_c \underline{t}^{\beta}$,

v a feasible solution of (IP) = $\min cx$ s.t. $Ax = b, x \geq 0$.

Output:

v^* an optimal solution of (IP).

Compute a reduced Gröbner basis G of I_A with respect to $>_c$

Compute $\underline{t}^{v^*} = \mathrm{redNF}(\underline{t}^{v}, G)$

return v^*

5 Computing I_A - The Algorithm of Hosten and Sturmfels

In the previous section, we have seen that we can solve our optimization problem as soon as we have a Gröbner basis of the toric ideal I_A associated to the constraint matrix.

Therefore, we will study how we can compute I_A as a saturation of the ideal I_{A_0} generated by the binomials induced by the vectors in a lattice basis of the kernel.

We start defining saturations and their properties in general.

5.1 Definition: Ideal Quotient, Saturation
Suppose $I, J \subseteq K[\underline{t}]$ are ideals. Then we define

$$I : J = \{f \in K[\underline{t}] \mid fJ \subseteq I\}$$

the **ideal quotient** of I and J and

$$I : J^\infty = \{f \in K[\underline{t}] \mid \exists n \in \mathbb{N} : fJ^n \subseteq I\}$$

the **saturation** of I and J.

We say that I is saturated with respect to J if $I : J = I$.

If $J = \langle g \rangle$ is principal, we may write $I : g$ (respectively $I : g^\infty$) instead of $I : J$ (respectively $I : J^\infty$).

5.2 Lemma
Suppose $I, K, J, J' \subseteq K[\underline{t}]$ are ideals.

a) $I : J$ is an ideal of $K[\underline{t}]$.

b) $I \subseteq I : J$

c) $(I : J) : K = I : (JK)$

d) If $K \subseteq I$, then $K : J \subseteq I : J$

e) If $J' \subseteq J$, then $I : J \subseteq I : J'$. In particular: $I : J \subseteq I : J^n$ for any $n \in \mathbb{N}$, $n > 0$.

f) a) - d) hold true for saturations as well.

g) Suppose $g \in K[\underline{t}]$. If there is an $f \in K[\underline{t}]$ such that $fg - 1 \in I$, then I is saturated with respect to g (that is: $I : g = I$).

Proof:

a) $0 \in I : J$, so $I : J$ is nonempty. If $fJ \subseteq I, gJ \subseteq I$, then $(f+g)J \subseteq I$ as well. If $fJ \subseteq I$ and $r \in K[\underline{t}]$, then $(rf)J = r(fJ) \subseteq I$ since I is an ideal.

b) $IJ \subseteq I$ since I is an ideal.

c) Suppose $f \in (I : J) : K \Leftrightarrow fK \in (I : J) \Leftrightarrow fKJ \in I \Leftrightarrow f \in I : (JK)$

d) Suppose $f \in K : J$, that is $fJ \subseteq K \subseteq I$, so $f \in I : J$.

e) Suppose $f \in I : J$, then $fJ \subset I$, but since $J' \subseteq J$, $fJ' \subseteq I$ as well, so $f \in I : J'$.

f)
$$I = I : J^0 \subseteq I : J \subseteq I : J^2 \subseteq I : J^3 \subseteq ...$$

is a chain of ideals in $K[\underline{t}]$. Since $K[\underline{t}]$ is noetherian, this chain will get stationary at some position, say at $I : J^k$. Therefore, it holds $I : J^\infty = I : J^k$. That means we can write a saturation as ideal quotient and we know that a) - d) hold for ideal quotients.

In addition, this shows that our definition of being saturated makes sense: if $I = I : J$, then by induction also

$$I : J^\infty = I : J^k = (I : J) : J^{k-1} = I : J^{k-1} = ... = I.$$

g) Suppose $h \in (I : g)$, that is $hg \in I$. WE have $fhg \in I$ since I is an ideal and $fg - 1$ in I by assumption. It follows that $h = fgh - (fg-1)h \in I$, so we get $I : g = I$.

□

5.3 Lemma
Suppose $A \in \mathbb{Z}^{mxn}$ is a matrix with integer entries and I_A is the toric ideal associated to A. Then:

a) I_A is a prime ideal.

b) I_A is saturated with respect to t_i for all $i \in \{1, ..., n\}$.

Proof:

a)
An ideal $I \subseteq K[\underline{t}]$ is an prime ideal if and only if its quotient ring $K[\underline{t}]/I$ is an integral domain.

We have shown in the first statement of 4.12 that $I_A = \mathrm{Ker}(\phi)$ for the ring-homomorphism $\phi : K[\underline{t}] \to K[s_1, s_1^{-1}, ..., s_m, s_m^{-1}],\ t_i \mapsto s^{a_i}$. Using the homomorphy theorem for rings, $K[\underline{t}]/I = K[\underline{t}]/\mathrm{Ker}(\phi)$ is isomorphic to a subring R of $K[s_1, s_1^{-1}, ..., s_m, s_m^{-1}]$, which is not the ring $\{0\}$, since $\phi(1) = 1 \in R$

We can show using induction:
$K[s_1, s_1^{-1}, ..., s_m, s_m^{-1}]$
$\cong K[s_1, s_1^{-1}, ..., s_{m-1}, s_{m-1}^{-1}][s_m, s_m^{-}1]$
$\cong K[s_1, s_1^{-1}, ..., s_{m-1}, s_{m-1}^{-1}][a, b]/\langle ab - 1\rangle$
is an integral domain since $K[s_1, s_1^{-1}, ..., s_{m-1}, s_{m-1}^{-1}]$ is an integral domain (by induction) and $\langle ab - 1\rangle$ is a prime ideal (generated by an irreducible element).

Therefore, its subrings (but $\{0\}$) must be integral domains, too, so I_A is prime.

b)
Since $\phi(t_i) = s^\alpha \neq 0$ for some $\alpha \in \mathbb{Z}^n$, t_i is not in I_A for any $i \in \{1, ..., n\}$. Since I_A is prime, it follows from $t_i f \in I_A$ for some $f \in K[\underline{t}]$ that $f \in I_A$. By the definition of the ideal quotient, $I = I : t_i$ follows.

□

5.4 Theorem
Suppose $A \in \mathbb{Z}^{mxn}$ is a matrix with integer entries and $U = \{u_1, ..., u_r\}$ is a lattice basis for $\mathrm{Ker}(A)$.

$$I_A = \langle \underline{t}^{x^+} - \underline{t}^{x^-} | x \in \mathrm{Ker}(A)\rangle$$

$$I_{A_0} = \langle \underline{t}^{x^+} - \underline{t}^{x^-} | x \in U\rangle$$

Then:

$$I_A = I_{A_0} : \underline{t}^\infty = I_{A_0} : (t_1 * t_2 * ... * t_n)^\infty = (...((I_{A_0} : t_1^\infty) : t_2^\infty)...) : t_n^\infty$$

Proof:
All equalities but the first one are clear by definition respectively the by lemma 5.2.

Note that $I_{A_0} \subseteq I_A$ by definition, so using lemma 5.2, we get the "$\supseteq$"-direction of the first equality.

It remains to show the "$\subseteq$"-direction of the first equality. Let f be a generator of I_A, that is: there is an $x \in \mathrm{Ker}(A)$ with $f = \underline{t}^{x^+} - \underline{t}^{x^-}$.
Since U was a lattice basis of $\mathrm{Ker}(A)$, we may write $x = \sum_{k=1}^{r} z_k u_k$ for some $z_k \in \mathbb{Z}$.

We may write
$\underline{t}^{x} - 1$
$= \underline{t}^{x^+ - x^-} - 1$
$= \underline{t}^{\sum_{k=1}^{r} z_k u_k} - 1$
$= \underline{t}^{\sum_{k=1}^{r} (z_k u_k)^+ - (z_k u_k)^-} - 1$
$= \frac{\prod_{k=1}^{r} \underline{t}^{(z_k u_k)^+}}{\prod_{k=1}^{r} \underline{t}^{(z_k u_k)^-}} - 1$
as an expression in the ring of Laurent-polynomials $K[t_1, t_1^{-1}, ..., t_n, t_n^{-1}]$, where negative exponents are allowed.

Define $g = \underline{t}^{(z_1 u_1)^-} * ... * \underline{t}^{(z_r u_r)^-}$ as the denominator in the equation above, a monomial in $K[\underline{t}]$.

If we multiply the above equation with $g\underline{t}^{x^-}$, we get the following equation in $K[\underline{t}]$:
$g\underline{t}^{x^-}(\underline{t}^{x} - 1)$
$= g\underline{t}^{x^-}(\underline{t}^{x^+ - x^-} - 1)$
$= \underline{t}^{x^-} g\Big(\frac{\prod_{k=1}^{r} \underline{t}^{(z_k u_k)^+}}{g} - 1\Big)$
$= \underline{t}^{x^-}\Big(\prod_{k=1}^{r} \underline{t}^{(z_k u_k)^+} - g\Big)$
$= \underline{t}^{x^-}\Big(\prod_{k=1}^{r} \underline{t}^{(z_k u_k)^+} - \prod_{k=1}^{r} \underline{t}^{(z_k u_k)^-}\Big).$

On the other hand, we get:
$g\underline{t}^{x^-}(\underline{t}^{x} - 1)$
$= g\underline{t}^{x^-}(\underline{t}^{x^+ - x^-} - 1)$
$= g(\underline{t}^{x^+ - x^- + x^-} - \underline{t}^{x^-})$
$= g(\underline{t}^{x^+} - \underline{t}^{x^-})$
$= gf.$

If we could show $d = \prod_{k=1}^{r} \underline{t}^{(z_k u_k)^+} - \prod_{k=1}^{r} \underline{t}^{(z_k u_k)^-} \in I_{A_0}$ (and thus $\underline{t}^{x^-} * d \in I_{A_0}$, too), we would have $gf \in I_{A_0}$, and thus $f \in I_{A_0} : \underline{t}^{\infty}$, since g is a monomial.

Define
$$d_j = \prod_{k=1}^{j} \underline{t}^{(z_k u_k)^+} - \prod_{k=1}^{j} \underline{t}^{(z_k u_k)^-}$$
for $j \in \{1, ..., r\}$. Show $d_j \in I_{A_0}$ for all j by induction, then $d = d_r \in I_{A_0}$ follows.

j = 1:

$d_1 = \underline{t}^{(z_1u_1)^+} - \underline{t}^{(z_1u_1)^-}$. If $z_1 = 0$, then $d_1 = 0 \in I_{A_0}$, so suppose $z_1 \neq 0$.

Set $\sigma_k = \begin{cases} 1 & , z_k > 0, \\ -1 & , z_k < 0. \end{cases}$

We have

$(z_ku_k)^+ = \begin{cases} |z_k|u_k^+ & , \sigma_k = 1, \\ |z_k|u_k^- & , \sigma_k = -1, \end{cases}$

and

$(z_ku_k)^- = \begin{cases} |z_k|u_k^- & , \sigma_k = 1, \\ |z_k|u_k^+ & , \sigma_k = -1. \end{cases}$

Therefore

$d_1 = \underline{t}^{(z_1u_1)^+} - \underline{t}^{(z_1u_1)^-}$
$= \sigma_1\big(\underline{t}^{|z_1|u_1^+} - \underline{t}^{|z_1|u_1^-}\big)$
$= \sigma_1\big((\underline{t}^{u_1^+})^{|z_1|} - (\underline{t}^{u_1^-})^{|z_1|}\big)$ and by multiplying out
$= \sigma_1\Big(\underline{t}^{u_1^+} - \underline{t}^{u_1^-}\Big)\Big(\sum_{s=0}^{|z_1|-1} (\underline{t}^{u_1^+})^s - (\underline{t}^{u_1^-})^{|z_1|-1-s}\Big).$

Since $(\underline{t}^{u_1^+} - \underline{t}^{u_1^-}) \in I_{A_0}$ and I_{A_0} is an ideal, the whole product $d_1 \in I_A$.

j > 1:

Suppose $d_{j-1} \in I_{A_0}$ by induction and $z_j \neq 0$ (since otherwise $d_j = d_{j-1} \in I_{A_0}$).

$d_j = \prod_{k=1}^{j} \underline{t}^{(z_ku_k)^+} - \prod_{k=1}^{j} \underline{t}^{(z_ku_k)^-}$
$= \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^+}\Big) - \prod_{k=1}^{j} \underline{t}^{(z_ku_k)^-}$ and by adding 0 in a nice way
$= \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^+} - \underline{t}^{(z_ju_j)^-}\Big) - \prod_{k=1}^{j} \underline{t}^{(z_ku_k)^-} + \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^-}\Big)$
$= \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^+} - \underline{t}^{(z_ju_j)^-}\Big) - \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^-}\Big)\Big(\underline{t}^{(z_ju_j)^-}\Big) + \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^-}\Big)$
$= \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^+} - \underline{t}^{(z_ju_j)^-}\Big) + \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+} - \prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^-}\Big)\Big(\underline{t}^{(z_ju_j)^-}\Big)$
$= \Big(\prod_{k=1}^{j-1} \underline{t}^{(z_ku_k)^+}\Big)\Big(\underline{t}^{(z_ju_j)^+} - \underline{t}^{(z_ju_j)^-}\Big) + d_{j-1}\Big(\underline{t}^{(z_ju_j)^-}\Big)$

The first summand is in I_{A_0}, since we may show $\underline{t}^{(z_ju_j)^+} - \underline{t}^{(z_ju_j)^-} \in I_{A_0}$ with the same arguments as in the case $j = 1$.
The second summand is in I_{A_0} since it is a multiple of d_{j-1} and $d_{j-1} \in I_{A_0}$ by induction.
Therefore, the $d_j \in I_{A_0}$ for all $j \in \{1, ..., r\}$ and thus $f \in I_{A_0} : \underline{t}^\infty$. Since f was chosen as an arbitrary generator, $I_A \subseteq I_{A_0} : \underline{t}^\infty$ follows. □

With this knowledge, we are able to compute I_A.

For example, we know from the proof of lemma 5.2 that we can write I_A as ideal quotient $I_A = I_{A_0} : \underline{t}^k$ for some $k \in \mathbb{N}$, so we could apply methods from computer algebra to compute this ideal quotient (see [Bö, p. 78]).

However, we can find much more efficient methods for our special situation. Depending on several properties of the integer program, there are different algorithms to compute the saturation, which are presented in Christine Theiss' diploma thesis [The99].

In the rest of this section, we will show how to compute the saturation in a certain simple situation where the ideal is homogeneous, using an algorithm given by Serkan Hosten and Bernd Sturmfels in [HS95].

5.5 Definition: Homogeneous

We call a polynomial $f \in K[\underline{t}]$ **homogeneous** with respect to some weight vector $w \in \mathbb{R}^n$ if all terms of f have the same degree $\deg_w(f)$.

We call an ideal $I \subseteq K[\underline{t}]$ **homogeneous** with respect to some weight vector $w \in \mathbb{R}^n$ if we can write $I = \langle H \rangle$ for some set $H \subseteq K[\underline{t}]$ of homogeneous polynomials.

Next, I will show some properties of homogeneous ideals, especially that they behave well with respect to the algorithms introduced in the third section.

5.6 Lemma

An ideal $I \subseteq K[\underline{t}]$ is homogeneous (with respect to some weight vector $w \in \mathbb{R}^n$) if and only if for every $f \in I$, its homogeneous components lie in I.

Proof:

"=>"

Suppose H is a set of homogeneous generators of I. We may write $f = \sum_{i=1}^{k} a_i h_i$ for some $h_i \in H$, $a_i \in K[\underline{t}]$ where $a_i h_i \in I$. If we split a_i into its homogeneous components a_{ij} $(j \in \{1, \ldots, l\})$, we have $a_{ij} h_i \in I$. Since the homogeneous components of f are sums of the $a_{ij} h_i$ with the same degree, these are in I, too.

"<="

Suppose for every $f \in I$ its homogeneous components are in I. Then

$$H = \{h | h \text{ is a homogeneous component of some } f \in I\}$$

is a set of homogeneous generators of I, so I is homogeneous.

□

5.7 Proposition

Let $>$ be a global monomial ordering and $w \in \mathbb{R}^n$ a weight vector.

a) Suppose f and all elements of $G = \{g_1, \dots, g_q\} \subset K[\underline{t}]$ are homogeneous (with respect to w). Then $\text{redNF}(f, G)$ is homogeneous, too.

b) Suppose f and $g \in K[\underline{t}]$ are homogeneous, then $\text{spoly}(f, g)$ is homogeneous.

c) Suppose $I \subseteq K[\underline{t}]$ is a homogeneous ideal, then its reduced Gröbner basis consists of homogeneous elements.

Proof:

a)

Applying algorithm 3.12 gives us a representation

$$f = \sum_{i=1}^{k} a_i g_{j_i} + r$$

where $j_i \in \{1, \dots, q\}$, $a_i = \frac{\text{LT}(f)}{\text{LT}(g_{j_i})}$, $r = \text{redNF}(f, G)$ and k is the number of times the if-step occurs.

Suppose f is homogeneous of degree $a \in \mathbb{R}$ and g_i is homogeneous of degree $b_i \in R$. Then $a_i * g_{j_i}$ is homogeneous of degree $a = (a - b_{j_i}) + b_{j_i}$ for all $i \in \{1, \dots, q\}$. Since we may write $r = f - \sum_{i=1}^{k} a_i g_{j_i}$ and all summands occuring on the right hand side are homogeneous of the same degree a, $r = \text{redNF}(f, G)$ is also homogeneous of this degree.

b)

Suppose $\deg_w(f) = a \in \mathbb{R}$ is the degree of f (and all its terms, since it is homogeneous), $\deg_w(g) = b \in \mathbb{R}$, $deg_w(\text{lcm}(\text{LM}(f), \text{LM}(g))) = c \in R$. Then both parts of $\text{spoly}(f, g) := \frac{\text{lcm}(\text{LM}(f),\text{LM}(g))}{\text{LT}(f)} f - \frac{\text{lcm}(\text{LM}(f),\text{LM}(g))}{\text{LT}(g)} g$ are homogeneous of degree $c = (c - a) + a = (c - b) + b$.

c)

Suppose I is a homogeneous ideal. Since $K[\underline{t}]$ is noetherian, we may find a finite set of generators G with $I = \langle G \rangle$. Similar to the proof of lemma 5.6, we may get a finite set of homogeneous generators H by defining

$$H = \{h | h \text{ is a homogeneous component of some } g \in G\}$$

Using this set as starting set for the Buchberger algorithm 3.22, we know from a) and b) that we get a Gröbner basis consisting of homogeneous elements.

If we use the procedure from the proof of theorem 3.18, we see with a) that the reduced Gröbner basis of I consists of homogeneous elements.

□

5.8 Lemma
Let w be a weight vector such that $I \subseteq K[\underline{t}]$ is a homogeneous ideal and g is a homogeneous element. Then $I : g$ and $I : g^\infty$ are homogeneous ideals, too.

Proof:
Suppose $f \in I : g$, that is $fg \in I$. Since I is homogeneous, all homogeneous components of fg lie in I (see lemma 5.6). Since g is homogeneous, the homogeneous components of fg are exactly the homogeneous components of f multiplied with g.
It follows that the homogeneous components of f lie in $I : g$, so $I : g$ is a homogeneous ideal. Since $I : g^\infty = I : g^k$ for some $k \in \mathbb{N}$ (as in the proof of lemma 5.2) and g^k is a homogeneous element, $I : g^\infty$ is homogeneous with respect to the same weight vector. □

The following theorem shows, that we can indeed compute the saturation in a nice way if I_{A_0} is homogeneous.

5.9 Theorem
Suppose $I \subseteq K[\underline{t}]$ is an ideal which is homogeneous with respect to some positive weight vector $w \in \mathbb{R}^n_{>0}$. Let $i \in \{1, ..., n\}$ be arbitrary. Let $>_{ls}$ be the local reverse lexicograpical ordering with respect to the variable ordering $t_i < t_1 < ... < t_{i-1} < t_{i+1} < ... < t_n < 1$ (see example 3.5).
Let $>_w$ be the global monomial ordering which is induced by w with $>_{ls}$ as tie-break (see example 3.7).
Let G be the reduced Gröbner basis of I with respect to $>_w$.
Then the set

$$G' = \{g \in K[\underline{t}] \mid t_i \nmid g, \exists k \in \mathbb{N} : t_i^k g \in G\}$$

which is obtained by dividing all elements of G by the highest possible power of t_i is a Gröbner basis of $I : t_i^\infty$ with respect to $>_w$.

Proof:
Suppose $g \in G'$, then there is a $k \in \mathbb{N}$ with $gt_i^k \in G \subset I$, so G' is indeed a finite subset of $I : t_i^\infty$ and $L(G') \subseteq L(I : t_i^\infty)$.
We still have to show:

$$L(I : t_i^\infty) \subseteq L(G')$$

So suppose $f \in I : t_i^\infty$, so there is some $k \in \mathbb{N}$ with $f t_i^k \in I$.
Since G is a Gröbner basis of I, there is some $g \in G$ such that $\mathrm{LM}(g)$ divides $\mathrm{LM}(f t_i^k) = \mathrm{LM}(f) * t_i^k$.
Since G is the reduced Gröbner basis of I, g is homogeneous with respect to w (as shown in lemma 5.8). Therefore, $\mathrm{LM}_{>_w}(g) = \mathrm{LM}_{>_{ls}}(g)$, so by the definiton of $>_{ls}$ the leading monomial of g is a monomial such that t_i occurs with a minimal power, say $q \in \mathbb{N}$.

By the definiton of G', $\frac{g}{t_i^q} \in G'$ and $\mathrm{LM}(\frac{g}{t_i^q}) = \frac{LM(g)}{t_i^q}$ divides $\mathrm{LM}(f t_i^k) = \mathrm{LM}(f) * t_i^k$.

By the choice of q, $\mathrm{LM}(\frac{g}{t_i^q})$ does not contain a power of t_i, so we have $\mathrm{LM}(\frac{g}{t_i^q})$ divides $\mathrm{LM}(f)$.

It follows $LM(f) \in L(G')$, so G' is indeed a Gröbner basis of $I : t_i^\infty$. □

The theorem above allows us to show the assumption 4.14 needed for the correctness of the general algorithm, if the ideal is homogeneous.

5.10 Theorem
Let $A \in \mathbb{Z}^{mxn}$, $>$ any global monomial ordering.

Suppose I_{A_0} is homogeneous with respect to some positive weight vector $w \in \mathbb{R}^n_{>0}$.

Then the reduced Gröbner basis of I_A with respect to $>$ consists of primitive binomials.

Proof:
Let $>_{ls,i}$ be the local reverse lexicograpical ordering from example 3.5 with respect to the variable ordering $t_i < t_1 < \ldots < t_{i-1} < t_{i+1} < \ldots < t_n < 1$. Let be $>_{w,i}$ the monomial ordering induced by w with $>_{ls,i}$ as tie-break.

I_{A_0} is by definition generated by finitely many primitive binomials, so we can apply the Buchberger-algorithm to these generators to get a reduced Gröbner basis with respect to $>_{w,1}$ consisting of pure binomials according to proposition 4.4.
Using theorem 5.9, we get again a Gröbner basis of $I : t_1^\infty$ with respect to $>_{w,1}$ consisting of pure binomials since we only divide both monomials in each generator by a power of t_1.

We can apply the Buchberger algorithm (and reduction) again to compute a reduced Gröbner basis of $I : t_1^\infty$ with respect to $>_{w,2}$, which consists again of pure binomials.

Since $I : t_1^\infty$ is homogeneous again, we may iterate this process to get a Gröbner basis of I_A with respect to $>_{w,n}$ consisting of finitely many pure binomials. We can use these generators to compute a reduced Gröbner basis G of I_A with respect to $>$ consisting of pure binomials.

Suppose the non-primitive binomial $f = \underline{t}^x \underline{t}^v - \underline{t}^x \underline{t}^u \in I_A$ (where $x \neq 0$ and $\underline{t}^v$ and $\underline{t}^u$ are coprime), then the primitive binomial $f' = \frac{f}{\underline{t}^x} \in I_A : \underline{t}^x \subseteq I_A : \underline{t}^\infty = I_A$.
By the definition of a Gröbner basis, there is a element $g \in G$ such that $\mathrm{LT}(g)$ divides $LT(f')$ and thus divides $\mathrm{LT}(f)$ strictly, that is $\langle LT(f)\rangle \subsetneq \langle LT(g)\rangle$. Since a reduced Gröbner basis is minimal, $f \notin G$.

This holds true for all non-primitive binomials, so G consists only of primitive ones. □

Before we state the final algorithm, we want to see how we can express the condition of I_A being homogeneous as a property of the integer program.

5.11 Proposition
Suppose $A \in \mathbb{Z}^{mxn}$. The following statements are equivalent:

a) There is a positive weight vector $w \in \mathbb{R}^n_{>0}$ such that I_{A_0} is homogeneous with respect to w.

b) There is a positive weight vector $gw \in \mathbb{R}^n_{>0}$ such that I_A is homogeneous with respect to w.

c) There is a vector $w \in \mathbb{R}^n_{>0}$ in the row-space of A (the $\mathbb{R}$ vector space $V \subseteq \mathbb{R}^n$ spanned by the rows of A), which has only positive components.

Proof:
Note that the row-space of a matrix is the orthogonal complement of its kernel, so we have that w is in the row-space of A if and only if $w^T x = 0$ for all $x \in \mathrm{Ker}(A)$, see [Str03, p. 187].

We show b) ⇔ c):

c) => b)

Suppose $w \in \mathbb{R}^n$ is a vector in the row-space of A with $w > 0$ component-wise and $x \in \mathrm{Ker}(A)$. Then: $0 = w^T x = w^T(x^+ - x^-)$. Hence $w^T x^+ = w^T x^-$. It follows, that $\deg_w(\underline{t}^{x^+}) = \deg_w(\underline{t}^{x^-})$, so the generators of I_A are homogeneous with respect to w; I_A is a homogeneous ideal.

b) => c)

We show that no monomial is contained in I_A and use the lemma about homogeneous ideals to show that the weight vector is in the row-space.

Look at I_0, the toric ideal associated to the zero-matrix 0.
Note that the unit vectors span Ker(0), so we may set

$$I := I_{0_0} = \langle t_i - 1 | i \in \{1, ..., n\} \rangle$$

Since $t_i * 1 - 1 \in I$, I is saturated with respect to every variable (see lemma 5.2), so by theorem 5.4, $I = I_0$.

Since we have shown, that I_0 is a prime ideal and a prime ideal is a strict subset of $K[\underline{t}]$, $1 \notin I$.

Suppose some monomial $\underline{t}^\alpha \in I$. Then $1 \in I : \underline{t}^\alpha \subseteq I : \underline{t}^\infty = I$ by lemma 5.3. Hence, no monomial is contained in I.

Since $\text{Ker}(A) \subseteq \text{Ker}(0)$, we also have $I_A \subseteq I_0$, so there is no monomial in I_A.

Suppose w defines a positive weight vector such that I_A is homogeneous with respect to w. If $f = \underline{t}^{x^+} - \underline{t}^{x^-}$ would not be homogeneous with respect to w for some $x \in \text{Ker}(A)$, then its homogeneous components, the monomials $\underline{t}^{x^+}$ and $\underline{t}^{x^-}$, would lie in I_A (see lemma 5.6). This is not possible since I_A contains no monomials.

Hence, we have $0 = w^T x = w^T(x^+ - x^-)$ since $w^T x^+ = w^T x^-$ for all $x \in \text{Ker}(A)$, so w is in the row space.

Now we can use this equivalency to show a) ⇔ c):

c) => a)

Since the generators of I_{A_0} are a subset of the generators of I_A, the statement follows with the same proof as in c) => b).

a) => c)

Suppose w is a positive weight vector such that I_{A_0} is homogeneous with respect to w. By applying theorem 5.9 iteratively as in the proof of theorem 5.10, we can show that I_A is homogeneous with respect to the same weight vector, so we are in the situation of b) and can apply the proof of b) => c).

$\square$

5.12 Algorithm: Hosten and Sturmfels

Input:

$A \in \mathbb{Z}^{m x n}$,

$w \in \mathbb{R}^n_{>0}$ a vector in the row-space of A,

$c \in \mathbb{R}^n_{\geq 0}$,

$>_c$ a global monomial ordering with $c^T\alpha > c^T\beta \Rightarrow \underline{t}^\alpha >_c \underline{t}^\beta$,

v a feasible solution of (IP) = min cx s.t. $Ax = b, x \geq 0$.

Output:

v^* an optimal solution of (IP).

Let $>_{ls,i}$ be the local reverse lexicographical ordering w.r.t the variable ordering

$t_i < t_1 < \ldots < t_{i-1} < t_{i+1} < \ldots < t_n$

Let $>_{w,i}$ be the global monomial ordering induced by w with $>_{ls,i}$ as tie-break.

1 Compute a lattice basis U of Ker(A)

2 $I_{A_0} := \langle \underline{t}^{x^+} - \underline{t}^{x^-} | x \in U \rangle$

3 Compute a reduced Gröbner basis G_0 of I_{A_0} with respect to $>_{w,1}$

4 **for** $i = 1, \ldots, n-1$ **do**

5 $\quad G_i := \{g \in K[\underline{t}] \mid t_i \nmid g, \exists k \in \mathbb{N} : t_i^k g \in G_{i-1}\}$

6 $\quad I_{A_i} := \langle G_i \rangle$

7 $\quad$ Replace G_i by a reduced Gröbner basis of I_{A_i} with respect to $>_{w,i+1}$

8 **end**

9 $G_n := \{g \in K[\underline{t}] \mid t_n \nmid g, \exists k \in \mathbb{N} : t_n^k g \in G_{n-1}\}$

10 $I_A := \langle G_n \rangle$

11 Use G_n to compute a reduced Gröbner basis G of $I_A = I_{A_n}$ with respect to $>_c$

12 Compute $\underline{t}^{v^*} = \text{redNF}(\underline{t}^v, G)$

13 **return** v^*

Proof of termination:

We have seen in the third section, that computing a (reduced) Gröbner basis starting with a finite set of generators can be done in finite time (see algorithm 3.22 and theorem 3.18).

In particular, the reduction in step 12 can also be done in finite time.

Computing a lattice basis can be done in polynomial time using the LLL algorithm, see remark 4.8. $\square$

Proof of correctness:

We have shown in the proof of proposition 5.11, that if we suppose the existence of a positive vector in the row-space, I_{A_0} is homogeneous with respect to the weighted degree given by this vector, so we can iteratively apply theorem 5.9, as done in the proof of theorem 5.10.

Theorem 5.4 shows that we get indeed a Gröbner basis for I_A after the 10^{th} step, so the correctness follows by the correctness of the generic algorithm (see theorem 4.16), since we are in a situation where the assumptiom 4.14 holds. □

5.13 Example

If we consider the integer program (IP) from example 2.9 and the toric ideal associated to its constraint matrix $I_{A_0} = \langle g \rangle$ with $g = t_1 t_4 - t_2 t_3$ from example 4.11, the steps 4 up to 11 of the algorithm above become trivial since I_{A_0} is principal so the Gröbner basis of I_{A_0} with respect to any monomial ordering is $\{g\}$ (see example 3.23) and g is a primitive binomial, so we cannot divide both monomials by any variable.

We have already computed in example 3.16 that $\text{redNF}(\underline{t}^v, \{g\}) = t_1^{10} t_2^{40} t_3^{30}$, where $v = (40, 10, 0, 30)^T$ is the initial solution of (IP) given in example 2.9.

Therefore, $v^* = (10, 40, 30, 0)^T$ is an optimal solution for (IP).

At the end of this thesis, we will take a look at the restrictions on the integer program we have in the final algorithm, and show that some of them can be eased.

5.14 Remark

First note that we really need the vector $w \in \mathbb{R}^n_{>0}$ in the row space, since otherwise our ideal would not be homogeneous and its components must be positive, since otherwise the monomial orderings we use would not be global. If there is no such vector, we may use other algorithms described in [The99] which work by introducing additional variables.

Finding such a vector (under the assumption of its existence) can be done using methods from linear optimization:

Instead of solving the system of strict inequalities

$$\sum_{i=1}^{m} a^i z_i > 0$$

(where a^i denotes the i^{th} row of A), we can also solve the system of $\geq$-inequalities

$$\sum_{i=1}^{m} a^i z_i \geq 1$$

since if $z \in \mathbb{R}^m$ solves the first system, we can find $\lambda \in \mathbb{N}$ such that λz solves the second system.

Note that the second system looks like a constraint of a linear program as in definiton 2.1, so we may apply methods for finding initial solutions of linear programs to find a solution z and thus $w = A^T z$, a positive vector in the row space of A.

5.15 Remark

In the case $c \in \mathbb{R}^n_{\geq 0}$ as in the algorithm above, it is easy to find a global monomial ordering $>_c$ as desired, we can use the monomial ordering induced by c with $>_{dp}$ as tie-break as in example 3.7.

But we could indeed drop this condition using more theory from optimization: We can also solve the problem in the case $c \notin \mathbb{R}^n_{\geq 0}$ if our problem has an optimal solution (this is always the case for $c \in \mathbb{R}^n_{\geq 0}$ if the program has a solution at all as shown in 2.8). In this situation, we can use efficient methods from linear optimization to solve the linear relaxation of (IP) (the linear program we get by dropping the condition $x \in \mathbb{Z}^n$).
We can derive a vector a vector $z \in \mathbb{N}^n$ from the optimal solution of this linear program such that any monomial ordering $>_z$ induced by z is compatible with (IP) (that is $c^T x > c^T y \Rightarrow \underline{t}^x >_z \underline{t}^y$ for feasible solutions x, y).
Since this monomial ordering is not compatible with all integer programs in $IP_{A,c}$, the reduced Gröbner basis of I_A would not be a test set, but the theory would work for the special instance $IP_{A,c}(b)$ we used to define $>_z$.
For more information on how to find this vector, see [The99, p. 43f].

5.16 Remark

We require an initial solution of the integer program as input to our algorithm.

In many real-life applications of integer programming, it is easy to give such an non-optimal but feasible initial solution.

If we don't have an initial solution given, we can compute one either by using methods from optimization or by using computer algebra: we can add variables $y_1, \ldots, y_m$ (where m is the number of constraints) to define a certain ideal related to the toric ideal I_A, compute a Gröbner basis of this new ideal and then reduce $\underline{y}^b$ with respect to this Gröbner basis as shown in [The99, p. 52] and [Stu96, p. 43].

Note that it may be more convenient to use the algorithm of Conti and Traverso instead of the algorithm of Hosten and Sturmfels in this setup.

Bibliography

[Bö] Böhm, Janko: *Computer Algebra*. – Lecture Notes (TU Kaiserslautern, winter term 2012/13), http://www.mathematik.uni-kl.de/~boehm/lehre/1213_CA/ca.pdf

[HS95] Hosten., S. ; Sturmfels, B.: GRIN: An implementation of Gröbner bases for integer prgramming. In: *Balas, E.: Integer programming and combinatorial optimization* (1995)

[Kar72] Karp, Richard M.: Reducibility Among Combinatorial Problems. In: *R. E. Miller and J. W. Thatcher (editors): Complexity of Computer Computations* (1972)

[Kar84] Karmarkar, Narendra: A New Polynomial Time Algorithm for Linear Programming. In: *Combinatorica* (1984)

[KM72] Klee, V. ; Minty, G.J.: How Good is the Simplex Algorithm? In: *Inequalities III* (1972)

[LLL82] Lenstra, A. K. ; Lenstra, H. W. ; Lovász, L.: Factoring polynomials with rational coefficients. In: *Mathematische Annalen* (1982)

[Mar] Markwig, Thomas: *Commutative Algebra*. – Lecture Notes (TU Kaiserslautern, winter term 2010/11), http://www.mathematik.uni-kl.de/~keilen/download/Lehre/MGSS09/CommutativeAlg.pdf

[Str03] Strang, Gilbert: *Introduction to Linear Algebra*. Wellesley-Cambridge Press,U.S., 2003

[Stu96] Sturmfels, Bernd: *Gröbner Bases and Convex Polytopes*. American Mathematical Soc., 1996

[The99] Theis, Christine: *Der Buchberger-Algorithmus fuer torische Ideale und seine Anwendung in der ganzzahligen Optimierung*, Universitaet des Saarlandes, Diplomarbeit, 1999