Journal of Algebra 324 (2010) 1098–1113

Contents lists available at ScienceDirect

Journal of Algebra

www.elsevier.com/locate/jalgebra

Normaliz: Algorithms for affine monoids and rational cones

Winfried Bruns [a,*], Bogdan Ichim [b,1]

[a] Universität Osnabrück, FB Mathematik/Informatik, 49069 Osnabrück, Germany
[b] Institute of Mathematics, C.P. 1-764, 010702 Bucharest, Romania

ARTICLE INFO

Article history:
Received 9 November 2009
Available online 11 March 2010
Communicated by Reinhard Laubenbacher

Keywords:
Affine monoid
Normalization
Rational cone
Hilbert basis

ABSTRACT

Normaliz is a program for the computation of Hilbert bases of rational cones and the normalizations of affine monoids. It may also be used for solving diophantine linear systems. In this paper we present the algorithms implemented in the program.

1. Introduction

The program Normaliz got its name from the first task for which it was designed: the computation of normalizations of affine monoids (or semigroups in other terminology). This task amounts to the computation of the Hilbert basis of the monoid of lattice points in a rational cone C with given generating system $x_1, \dots, x_n$ (see Section 2 for terminology and [2] for mathematical background). Such cones can be described equivalently by homogeneous linear diophantine equations and inequalities, and the computation of the normalization is equivalent to solving such systems.

The mathematical aspects of the first implementation of Normaliz have been documented in [8]. In this paper we present the algorithms that have been added or modified in version 2.0 and later. The user interface has been described in [6]. Further extensions, for example parallelization of time critical steps, are still experimental; they will be presented in [4].

As any other program that computes Hilbert bases, Normaliz first determines a system of generators of the monoid. Section 3 describes Normaliz' approach for the reduction of the system of generators to a Hilbert basis—often (but not always) the most time consuming part of the computation. Section 4 contains our implementation of the Fourier–Motzkin elimination, which is tuned for obtaining best results in the case when most of the facets are simplicial. (Fourier–Motzkin elimination

* Corresponding author.
E-mail addresses: wbruns@uos.de (W. Bruns), bogdan.ichim@imar.ro (B. Ichim).

[1] The second author was partially supported by CNCSIS grant RP-1 no. 7/01.07.2009 during the preparation of this work.

0021-8693/$ – see front matter
doi:10.1016/j.jalgebra.2010.01.031

computes the convex hull of a finite set of points, or, in homogenized form, the support hyperplanes of a finitely generated cone.) We need this variant for the new algorithm by which h-vector and Hilbert polynomial are computed. It is based on line shellings and will be presented in Section 6. Finally, our implementation of Pottier's algorithm [14] is presented in Section 7. In our interpretation, this "dual" algorithm is based on a representation of the cone as an intersection of halfspaces, whereas the "primal" algorithm of Normaliz starts from a system of generators.

The first version of Normaliz was a C program created by Winfried Bruns and Robert Koch in 1997–1998 and extended in 2003 by Witold Jarnicki. Version 2.0 (2007–2008) was completely rewritten in C++ by Bogdan Ichim. Pottier's algorithm for solving systems of inequalities and equations was added in version 2.1. Christof Söger enhanced the user interface in version 2.2, the currently public version. The distribution of Normaliz [5] contains a Singular library and a Macaulay2 package; for the latter, written by Gesa Kämpf, see [7]. Andreas Paffenholz provided a polymake interface to Normaliz [13].

We wish to thank all colleagues who have contributed to the development of Normaliz.

2. Affine monoids and their Hilbert bases

We use the terminology introduced as in [2], but for the convenience of the reader we recall some important notions. A *rational cone* $C \subset \mathbb{R}^d$ is the intersection of finitely many linear halfspaces $H_\lambda^+ = \{x \in \mathbb{R}^d\colon\ \lambda(x) \geqslant 0\}$ where λ is a linear form with rational coefficients (with respect to the standard basis of $\mathbb{R}^d$). By the theorem of Minkowski and Weyl (for example, see [2, 1.15]), we can require equivalently that C is of type $\mathbb{R}_+x_1 + \cdots + \mathbb{R}_+x_n$ with $x_i \in \mathbb{Q}^d$, $i = 1, \ldots, n$. In this case, $x_1, \ldots, x_n$ form a *system of generators* for C. If C can be generated by a linearly independent set of generators, we say that C is a *simplicial* cone. If $\dim C = d$, then the halfspaces in an irredundant representation of C as an intersection of halfspaces are uniquely determined, and the corresponding linear forms λ_i are called *support forms* of C, after they have been further specialized such that $\lambda_i(\mathrm{gp}(M)) = \mathbb{Z}$. If $\mathrm{gp}(M) = \mathbb{Z}^d$, the last condition amounts to the requirement that the λ_i have coprime integral coefficients. (Such linear forms are called *primitive*.) In the following all cones are rational, and we omit this attribute accordingly. A cone is *pointed* if $x, -x \in C$ implies $x = 0$.

An *affine monoid* M is finitely generated and (isomorphic to) a submonoid of a lattice $\mathbb{Z}^d$. By $\mathrm{gp}(M)$ we denote the subgroup generated by M, and by $\operatorname{rank} M$ its rank. The support forms $\sigma_1, \ldots, \sigma_s$ of the cone $\mathbb{R}_+M \subset \mathbb{R}M$ are called the *support forms of M*. They define the *standard map*

$$\sigma : M \to \mathbb{Z}_+^s, \qquad \sigma(x) = \big(\sigma_1(x), \ldots, \sigma_s(x)\big).$$

We introduce the *total degree* $\operatorname{tdeg} x$ by $\operatorname{tdeg} x = \sigma_1(x) + \cdots + \sigma_s(x)$. (In [2] the total degree is denoted τ.)

The *unit group* $\mathrm{U}(M)$ consists of the elements $x \in M$ for which $-x \in M$ as well. It is not hard to see that $x \in \mathrm{U}(M)$ if and only if $\sigma(x) = 0$ (see [2, 2.14]), in other words, if and only if $\operatorname{tdeg} x = 0$. (However, in general $\operatorname{tdeg} x = \operatorname{tdeg} y$ does *not* imply $x - y \in \mathrm{U}(M)$ since $x - y$ need not belong to M.) One calls M *positive* if $\mathrm{U}(M) = 0$.

An element $x \in M$ is *irreducible* if $x \notin \mathrm{U}(M)$ and a representation $x = y + z$ with $y, z \in M$ is only possible with $y \in \mathrm{U}(M)$ or $z \in \mathrm{U}(M)$.

In the next definition we extend the terminology of [2] slightly.

Definition 1. Let M be a (not necessarily positive) affine monoid. A subset H of M is a *system of generators modulo* $\mathrm{U}(M)$ if $M = \mathbb{Z}_+H + \mathrm{U}(M)$, and H is a *Hilbert basis* if it is minimal with respect to this property.

A Hilbert basis is necessarily finite since M has a finite system of generators. Moreover, every system of generators modulo $\mathrm{U}(M)$ contains a Hilbert basis. Often we will use the following criterion (see [2, 2.14]).

Proposition 2. *$H \subset M$ is a Hilbert basis if and only if it is a system of representatives of the nonzero residue classes of the irreducible elements modulo* $\mathrm{U}(M)$.

The Hilbert basis of a positive affine monoid is uniquely determined and denoted by $\operatorname{Hilb}(M)$.

Suppose N is an overmonoid of M. Then we call $y \in N$ *integral* over M if $ky \in M$ for some $k \in \mathbb{Z}$, $k > 0$. The set of elements of N that are integral over M form the *integral closure* $\widehat{M}_N$ of M in N; it is itself a monoid. The *normalization* $\bar{M}$ of M is its integral closure in $\operatorname{gp}(M)$, and if $M = \bar{M}$, M is called *normal*.

If M is normal, the case in which we are mainly interested, then M splits in the form $\mathrm{U}(M) \oplus \sigma(M)$ (see [2, 2.26]) and we can state:

Proposition 3. *Let M be a normal affine monoid with standard map $\sigma : M \to \mathbb{Z}_+^s$. Then $H \subset M$ is a Hilbert basis of M if and only if σ maps H bijectively onto a Hilbert basis of $\sigma(M)$.*

It is a crucial fact that integral closures of affine monoids have a geometric description (see [2, 2.22]):

Theorem 4. *Let $M \subset N$ be submonoids of $\mathbb{Q}^d$, and $C = \mathbb{R}_+ M$.*

(1) *Then $\widehat{M}_N = C \cap N$.*
(2) *If M and N are affine monoids, then $\widehat{M}_N$ is affine, too.*

The second statement of the theorem is (an extended version of) *Gordan's lemma*.

The program Normaliz computes Hilbert bases of monoids of type $C \cap L$ where C is a pointed rational cone specified either (i) by a system $x_1, \dots, x_n \in \mathbb{Z}^d$ or (ii) a system $\sigma_1, \dots, \sigma_s \in (\mathbb{R}^d)^*$ of integral linear forms, and L is a lattice that can be chosen to be either $\mathbb{Z}^d$ or, in case (i), $\mathbb{Z}x_1 + \cdots + \mathbb{Z}x_n$. We will simply say that Normaliz computes Hilbert bases of rational cones. If C is pointed and their is no ambiguity about the lattice L, then we simply write $\operatorname{Hilb}(C)$ for $\operatorname{Hilb}(C \cap L)$.

Once a system of generators of C is known (either from the input data or as a result of a previous computation), Normaliz reduces this computation to the *full-dimensional* case in which $\dim C = \operatorname{rank} L$ and introduces coordinates for the identification $L = \mathbb{Z}^{\operatorname{rank} L}$. The necessary coordinate transformations are discussed in [8, Section 2].

3. Reduction

All algorithms that compute Hilbert bases of rational cones cannot avoid to first produce a system of generators that is nonminimal in general. In a second, perhaps intertwined, step the system of generators is shrunk to a Hilbert basis. This approach is based on the following proposition. Let us say that $y \in M$ *reduces* $x \in M$ if $y \notin \mathrm{U}(M)$, $x \neq y$, and $x - y \in M$.

Proposition 5. *Let M be an affine monoid* (*not necessarily positive or normal*), *$E \subset M$ a system of generators modulo* $\mathrm{U}(M)$, *and $x \in E$. If x is reduced by some $y \in E$, then $E \setminus \{x\}$ is again a system of generators modulo* $\mathrm{U}(M)$.

Proof. Note that E contains a Hilbert basis. It is enough to show that $E \setminus \{x\}$ contains a Hilbert basis as well. If $x - y \notin \mathrm{U}(M)$, then x is reducible, and does not belong to any Hilbert basis, and if $x - y \in \mathrm{U}(M)$, we can replace x by y in any Hilbert basis $H \subset E$ to which x belongs. $\square$

The proposition shows that one obtains a Hilbert basis from a set E of generators modulo $\mathrm{U}(M)$ by (i) removing all units from E, and (ii) successively discarding elements x such that $x - y \in M$ for some $y \in E$, $x \neq y$. After finitely many reduction steps one has reached a Hilbert basis.

The difficult question is of course to decide whether $x \in \mathrm{U}(M)$ or $x - y \in M$. However, if $M = C \cap L$ with a rational cone $C \subset \mathbb{R}^d$, and a sublattice L of $\mathbb{Q}^d$, then this question is very easy to decide, once the support forms $\sigma_1, \dots, \sigma_s$ of C are known:

$$x - y \in M \iff x - y \in C \iff \sigma_i(x - y) \geqslant 0, \quad i = 1, \dots, s,$$

and

$$x \in \mathrm{U}(M) \quad \Longleftrightarrow \quad \sigma_i(x) = 0, \quad i = 1, \dots, s.$$

Therefore, if C is given by a set of generators, the necessity of reduction forces us to compute the support forms of C. Normaliz's approach to this task is discussed in Section 4.

It is very important for efficiency to make reduction as fast as possible. Normaliz uses the following algorithm. The elements forming a system of generators are inserted into a set E ordered by increasing total degree such that at the end of the production phase E contains at most one element from each residue class modulo $\mathrm{U}(M)$ (and no element from $\mathrm{U}(M)$). Let $E = \{x_1, \dots, x_m\}$. Then a Hilbert basis H is extracted from E. Initially, H is the set of elements of minimal total degree in E, say $H = \{y_1, \dots, y_u\} = \{x_1, \dots, x_u\}$. For $i = u+1, \dots, m$ the element x_i is compared to the dynamically extended and reordered list $H = \{y_1, \dots, y_n\}$ as follows:

(R1) for $j = 1, \dots, n$,
  (a) if $\operatorname{tdeg} x_i < 2 \operatorname{tdeg} y_j$, then x_i is appended to H as y_{n+1};
  (b) if $x_i - y_j \in M$, then H is reordered as $H = \{y_j, y_1, \dots, y_{j-1}, y_{j+1}, \dots, y_n\}$;
(R2) if (i) or (ii) does not apply for any j, then x_i is appended to H as y_{n+1}.

For the justification of this procedure, note that $x - y \in M$ for some y with $2 \operatorname{tdeg} y \leqslant \operatorname{tdeg} x$ if x is reducible. Therefore (R1)(a) can be applied, provided $2 \operatorname{tdeg} y_k > \operatorname{tdeg} x_i$ for all $k \geqslant j$. This holds since E is ordered by ascending degree, the fact that no element in H that follows y_j has been touched by the rearrangement in (R1)(b): only elements with $2 \operatorname{tdeg} y_j \leqslant \operatorname{tdeg} x_i$ have been moved, and $\operatorname{tdeg} x_i \leqslant \operatorname{tdeg} x_k$ for all $k \geqslant i$.

Remark 6. (a) The "darwinistic" rearrangement in (R1)(b) above has a considerable effect as all tests have shown. It keeps the "successful reducers" at the head of the list. Moreover, successive x_i are often close to each other (based on empirical evidence), so that y_j has a good chance to reduce x_{i+1} if it reduces x_i.

(b) Instead of tdeg one can use any other convenient positive linear form $\tau : M \mapsto \mathbb{R}_+$ with the property that $\tau(y) = 0 \Leftrightarrow y \in \mathrm{U}(M)$.

In Section 7 we will encounter a situation in which the subset E to which reduction is to be applied need not to be a system of generators. Then we call a subset E' an *auto-reduction* of E, if $E' \cap \mathrm{U}(M) = \emptyset$ and no element of E' is reduced by another one.

4. Computing the dual cone

Let $C \subset \mathbb{R}^d$ be a rational cone and $L \subset \mathbb{Z}^d$ a lattice. In order to perform the reduction of a system of generators of the normal monoid $C \cap L$ to a Hilbert basis, as discussed in the previous section, one must know the hyperplanes that cut out C from $\mathbb{R}^d$, or rather the integral linear forms defining them. These linear forms generate the dual cone C^* in $(\mathbb{R}^d)^*$.

Conversely, if C is defined as the intersection of halfspaces represented by a system of generators of C^*, then the use of an algorithm based on a system of generators of C makes it necessary to find such a system. Since $C = C^{**}$ (see [2, 1.16]), this amounts again to the computation of the dual cone: the passage from C to C^* and that from C^* to C can be performed by the same algorithm.

In the following we take the viewpoint that a full-dimensional pointed rational cone $C \subset \mathbb{R}^d$ is given by a system of generators E, and that the linear forms generating C^* are to be computed. Normaliz uses the well-known Fourier–Motzkin elimination for this task, however with a simplicial refinement that we will describe in detail.

Fourier–Motzkin elimination is an inductive algorithm. It starts from the zero cone, and then inserts the generators $x_1, \dots, x_n$ successively, transforming the support hyperplanes of $C' = \mathbb{R}_+ x_1 + \dots + \mathbb{R}_+ x_{n-1}$ into those of $C = \mathbb{R}_+ x_1 + \dots + \mathbb{R}_+ x_n$. The transformation is given by the following theorem; for example, see [2, pp. 11, 12].

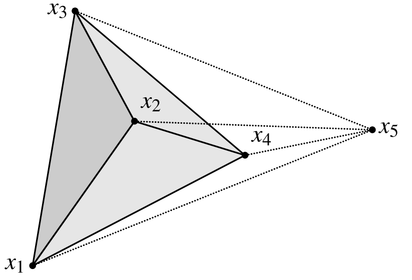

Fig. 1. Cross-section of extension of the cone.

Theorem 7. *Let C be generated by $x_1,\dots,x_n$ and suppose that $C'=\mathbb{R}_+x_1+\cdots+\mathbb{R}_+x_{n-1}$ is cut out by linear forms $\lambda_1,\dots,\lambda_m$. Let $\mathscr{P}=\{\lambda_i\colon \lambda_i(x_n)>0\}$, $\mathscr{N}=\{\lambda_i\colon \lambda_i(x_n)<0\}$, and $\mathscr{Z}=\{\lambda_i\colon \lambda_i(x_n)=0\}$. Then C is cut out by the linear forms in the set*

$$\mathscr{P}\cup\mathscr{Z}\cup\bigl\{\lambda_i(x_n)\lambda_j-\lambda_j(x_n)\lambda_i\colon \lambda_i\in\mathscr{P},\ \lambda_j\in\mathscr{N}\bigr\}.$$

In this raw form the algorithm produces $|\mathscr{P}|\cdot|\mathscr{N}|$ linear forms, from which the *new* facets have to be selected. While the complexity of this algorithm may seem negligible in view of the subsequent steps in the Hilbert basis computation, this is no longer so if applied in the computation of a shelling of C (see Section 6). But in the computation of a shelling the boundary of (a lifting of) C consists mainly of simplicial facets, and this allows an enormous acceleration. (The construction of the lifting ensures that most of its facets are simplicial; see Remark 12.)

In a geometric interpretation of Fourier–Motzkin elimination, we have to find the boundary V of that part of the surface of C' that is visible from x_n, or rather the decomposition of V into subfacets (faces of C of dimension $d-2$). Fig. 1 illustrates the inductive step of Fourier–Motzkin elimination in the three-dimensional cross-section of a four-dimensional cone. The area of the "old" cone visible from the "new" generator x_5 is the union of the cones spanned by the triangles $[x_1,x_4,x_2]$ and $[x_2,x_4,x_3]$, whereas V is the union of the cones over the line segments forming the cycle $[x_1,x_4,x_3,x_2,x_1]$.

Each subfacet S of C' is the intersection of two facets F and G and we call F and G *partners* with respect to S. In order to compute the new facets of C we have to find those subfacets S of C' whose two overfacets belong to $\mathscr{P}$ and $\mathscr{N}$, respectively. The new facets of C are then the cones $\mathbb{R}_+(S\cup\{x_n\})$.

Let E' be the subset of $E\setminus\{x_n\}$ whose elements are contained in a hyperplane belonging to $\mathscr{P}$ as well as in a hyperplane belonging to $\mathscr{N}$. Clearly, a facet F of C' can only contribute to a new facet of C if $|F\cap E'|\geqslant d-2$. While this observation is useful (and is applied), its effect is often rather limited.

Normaliz proceeds as follows; for simplicity we will identify subsets of E with the faces they generate.

(D1) It separates the facets in $\mathscr{P}$ and $\mathscr{N}$ into the subsets $\mathscr{P}_{\mathrm{simp}}$ and $\mathscr{N}_{\mathrm{simp}}$ of simplicial ones and the subsets $\mathscr{P}_{\mathrm{nonsimp}}$ and $\mathscr{N}_{\mathrm{nonsimp}}$ of nonsimplicial ones, discarding those facets that do not satisfy the condition $|F\cap E'|\geqslant d-2$.

(D2) All subfacets of all the facets $N\in\mathscr{N}_{\mathrm{simp}}$ are formed by simply taking the subsets S of cardinality $d-2$ of $N\cap E$ (which has cardinality $d-1$ in the simplicial case). The pairs (S,N) are stored in a set ordered by lexicographic comparison of the components S. In fact, if S appears with a second facet $N'\in\mathscr{N}_{\mathrm{simp}}$, then it cannot belong to V, and both pairs (S,N) and (S,N') can be discarded immediately. Forming the ordered set $\mathscr{T}$ is of complexity of order $q\log_2 q$ where $q=(d-1)|\mathscr{N}_{\mathrm{simp}}|$.

(D3) Each pair $(S,N)\in\mathscr{T}$ is compared to the facets in $G\in\mathscr{N}_{\mathrm{nonsimp}}\cup\mathscr{Z}$: if $S\subset G$, then the partner of F with respect to S does not belong to $\mathscr{P}$, and (S,N) can be deleted from $\mathscr{T}$. (In the critical situation arising from the computation of a shelling, the sets $\mathscr{N}_{\mathrm{nonsimp}}$ and $\mathscr{Z}$ are usually short.)

(D4) At this point $\mathscr{T}$ contains only pairs (S, N) such that the partner of N with respect to S indeed belongs to $\mathscr{P}$, and therefore gives rise to new facet. It remains to find the partners.

(D5) Normaliz now produces all subfacets S of the facets $P \in \mathscr{P}_{\text{simp}}$ and tries to find S as the first component of an element in the set $\mathscr{T}$. This search is of complexity of order $(d-1) \cdot |\mathscr{P}_{\text{simp}}| \cdot \log_2 q$, q as above.

If the search is successful, a new facet of C is produced, and the pair (S, N) is discarded from $\mathscr{T}$.

(D6) To find the partners in $\mathscr{P}_{\text{nonsimp}}$ for the remaining pairs (S, N) in $\mathscr{T}$, the sets S are compared to the facets $\mathscr{P}$ in $\mathscr{P}_{\text{nonsimp}}$. This comparison is successful in exactly one case, leading to a new facet.

(D7) Finally, the facets $N \in \mathscr{N}_{\text{nonsimp}}$ are paired with all facets $P \in \mathscr{P}$, as described in Theorem 7, and whether a hyperplane H produced is really a new facet of C is decided by the following rules, applied in the order given:
  - (i) if $|N \cap P \cap E| < d-2$, then H can be discarded;
  - (ii) if $|N \cap P \cap E| = d-2$ and $P \in \mathscr{P}_{\text{simp}}$, then H is a new facet;
  - (iii) H is a new facet if and only if $\operatorname{rank}(N \cap P \cap E) = d-2$;
  - (iv) (alternative to the rank test) H is a new facet if and only if the only nonsimplicial facets containing $N \cap P$ are N and P.

Which of the tests (iii) or (iv) is applied, is determined as follows: if the number of nonsimplicial facets is $< d^3$, then (iv) is applied, and otherwise the rank test is selected.

It is not hard to see that (iv) is sufficient and necessary for $N \cap P$ to have dimension $d-2$. Indeed, a subfacet is contained in exactly two facets, and if we have arrived at step (iv), $P \cap N$ cannot be contained in any simplicial facet G: since $P \cap N \geqslant d-2$, it must be a subfacet contained in G, and it would follow that $P = G$ or $N = G$, but both P and N are nonsimplicial.

Remark 8. (a) Computing the dual cone is essentially equivalent to computing the convex hull of a finite set of points: instead of the affine inhomogeneous system of inequalities we have to deal with its homogenization. Therefore one could consider other convex hull algorithms, like "gift wrapping" or "beneath and beyond" (see [12] for their comparison to Fourier–Motzkin elimination). The main advantages of Fourier–Motzkin elimination for Normaliz are that it does not require (but allows) the simultaneous computation of a triangulation, and furthermore that the incremental construction of C, adding one generator at a time, can be used very efficiently in some hard computations (see [4]).

(b) One can extend the idea of the simplicial refinement and work with a triangulation of the boundary of C' that is then extended to a triangulation of the boundary of C, accepting that a facet may decompose in many simplicial cones. In this way the pairing of "positive" and "negative" facets can be reduced to the creation of a totally ordered set and the search in such a set. In our tests the separate treatment of simplicial and nonsimplicial facets turned out superior.

5. The primal Normaliz algorithm

The *primal* algorithm of Normaliz proceeds as follows (after the initial coordinate transformation discussed in [8, Section 2]), starting from a system of generators $x_1, \dots, x_n$ of C:

(N1) the support hyperplanes of C are computed as described by Fourier–Motzkin elimination (Section 4);

(N2) intertwined with (N1), the lexicographic (or placing) triangulation of C is computed into which the generators $x_1, \dots, x_n$ are inserted in this order;

(N3) for each simplicial cone D in the triangulation $\operatorname{Hilb}(D \cap \mathbb{Z}^d)$ is determined;

(N4) the union of the sets $\operatorname{Hilb}(D \cap \mathbb{Z}^d)$ is reduced to $\operatorname{Hilb}(C)$.

After the completion of (N1) one knows C^* and can decide whether C is pointed since pointedness of C is equivalent to full-dimensionality of C^* (see [2, 1.19]).

In step (N2) the lexicographic triangulation Σ' of $C' = \mathbb{R}_+ x_1 + \cdots + \mathbb{R}_+ x_{n-1}$ is extended to a lexicographic triangulation Σ of $C = \mathbb{R}_+ x_1 + \cdots + \mathbb{R}_+ x_n$ as follows: Let $F_1, \dots, F_v$ be those facets of

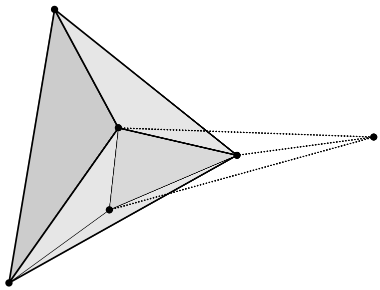

Fig. 2. Extension of triangulation.

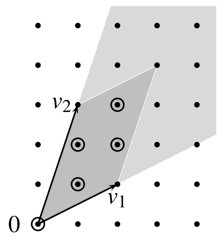

Fig. 3. The semi-open parallelotope and the set E.

the maximal cones in Σ' that lie in the facets of C visible from x_n: then $\Sigma = \Sigma' \cup \{F_i + \mathbb{R}_+ x_n\colon\ i = 1, \dots, v\}$. (If $x_n \in C'$, then $C = C'$ and $\Sigma = \Sigma'$.) (Compare [2, p. 267] for lexicographic triangulations.) This construction is illustrated by Fig. 2.

It only remains to explain how a set E_D of generators of $D \cap \mathbb{Z}^d$ is determined if D is simplicial, i.e., generated by a linearly independent set $V = \{v_1, \dots, v_d\} \subset \mathbb{Z}^d$. Following the notation of [2], we let

$$\operatorname{par}(v_1, \dots, v_d) = \{a_1 v_1 + \dots + a_d v_d\colon\ 0 \leqslant a_i < 1,\ i = 1, \dots, d\}$$

denote the semi-open parallelotope spanned by $v_1, \dots, v_d$. Then the set

$$E = E_D = \operatorname{par}(v_1, \dots, v_d) \cap \mathbb{Z}^d \tag{5.1}$$

generates $D \cap \mathbb{Z}^d$ as a free module over the free submonoid $\mathbb{Z}_+ v_1 + \dots + \mathbb{Z}_+ v_d$. In other words, every element $z \in D \cap \mathbb{Z}^d$ has a unique representation

$$z = x + \sum_{i=1}^{d} a_i v_i, \quad x \in E,\ a_i \in \mathbb{Z}_+. \tag{5.2}$$

See [2, 2.43] for this simple, but crucial fact. Clearly $E \cup \{v_1, \dots, v_d\}$ generates the monoid $D \cap \mathbb{Z}^d$. Fig. 3 illustrates the construction of E.

The efficient computation of E_d has been discussed in [8]; it amounts to finding a representative z for each residue class in $\mathbb{Z}^d / (\sum \mathbb{Z} v_i)$ and reducing it modulo $v_1, \dots, v_d$ to its representative in $\operatorname{par}(v_1, \dots, v_d)$.

In the following the attribute *local* refers to the simplicial cones D whereas *global* refers to C.

While the primal algorithm had been realized already in the first version of Normaliz (see [8]), it has now undergone several refinements.

Remark 9. (a) The lexicographic triangulation is used by Normaliz in its "normal" (meaning "standard") computation type. It is replaced by a shelling if the h-vector is to be computed (see Section 6).

(b) The total number of vectors generated by Normaliz is the sum of the multiplicities $\mu(D) = |\det(v_1, \dots, v_d)|$ of the simplicial cones D in the triangulation. If the monoid is defined by a lattice polytope P, then this number is the $\mathbb{Z}^d$-normalized volume of P, and therefore independent of the triangulation. (This count includes the zero vector in each simplicial cone; therefore the number of simplicial cones should be subtracted from the sum of multiplicities.)

6. *h*-vectors via shellings

For $N \subset \mathbb{R}^d$ we set

$$H_N(\boldsymbol{t}) = \sum_{x \in N \cap \mathbb{Z}^d} \boldsymbol{t}^x.$$

Here we use multi-exponent notation: $\boldsymbol{t}^x = t_1^{x_1} \cdots t_d^{x_d}$. The formal Laurent series is simply the "characteristic series" of the set $N \cap \mathbb{Z}^d \subset \mathbb{Z}^d$. If N is an "algebraic" object (for example, an affine monoid), then we can interpret $H_N(\boldsymbol{t})$ as the multigraded Hilbert series of N.

Let C be a cone and $M = C \cap \mathbb{Z}^d$. Suppose that C is triangulated by the conical complex $\mathscr{C}$, the standard situation in the primal algorithm of Normaliz. If $C_1, \dots, C_m$ are the maximal cones in $\mathscr{C}$, then

$$H_M(\boldsymbol{t}) = \sum_{i=1}^{m} H_{D_i}(\boldsymbol{t}), \quad D_i = C_i \setminus (C_1 \cup \cdots \cup C_{i-1}),\ i = 1, \dots, m. \tag{6.1}$$

Within C_i, the set D_i, $i = 1, \dots, m$, is the complement of the union of the sets of faces of $C_1, \dots, C_{i-1}$. Therefore it is the union of the (relative) interiors of those faces of C_i that are not contained in $C_1 \cup \cdots \cup C_{i-1}$. In order to compute $H_{D_i}(\boldsymbol{t})$, one has to solve two problems: (i) to compute $H_{\operatorname{int} D}(\boldsymbol{t})$ for a simplicial cone D, and (ii) to find the decomposition of D_i as a union of interiors of faces of C_i.

As in Section 5 we denote the linearly independent generators of the simplicial cone D by $v_1, \dots, v_d$ and consider the system of generators $E = E_D$. For a subset Y of $V = \{v_1, \dots, v_d\}$, let

$$\mathscr{H}_Y(\boldsymbol{t}) = H_{\mathbb{Z}_+ Y}(\boldsymbol{t}) = \prod_{v_i \in Y} \frac{1}{1 - \boldsymbol{t}^{y_i}} \quad \text{and} \quad \boldsymbol{t}^Y = \prod_{v_i \in Y} \boldsymbol{t}^{v_i}.$$

By definition, $\mathscr{H}_Y(\boldsymbol{t})$ is the Hilbert series of the free monoid generated by Y. In view of Eq. (5.2) one obtains

$$H_D(\boldsymbol{t}) = \mathscr{H}_V(\boldsymbol{t}) \sum_{x \in E} \boldsymbol{t}^x.$$

Now problem (i) is easily solved (compare [2, p. 234]):

$$H_{\operatorname{int}(D)} = (-1)^{\dim D} H_D\left(\boldsymbol{t}^{-1}\right) = \mathscr{H}_V(\boldsymbol{t}) \sum_{x \in E} \boldsymbol{t}^{v_1 + \cdots + v_d - x}.$$

Problem (ii) is very hard for an arbitrary order of the cones C_i in the triangulation. However, it becomes easy if $C_1, \dots, C_m$ is a shelling. Shellings are the classical tool for the investigation of h-vectors, as demonstrated by McMullen's proof of the upper bound theorem (see [3] or [15]). We need the notion of shelling only for complexes of simplicial cones (or polytopes), for which it reduces to a purely combinatorial condition.

Definition 10. Let $\mathscr{C}$ be a complex of simplicial cones (or polytopes) whose maximal cones have constant dimension d. An order $C_1, \dots, C_m$ of the maximal cones in $\mathscr{C}$ is called a *shelling* if $C_i \cap (C_1 \cup \dots \cup C_{i-1})$ is a union of facets of C_i for all i.

The next lemma solves problem (ii) for a shelling. For a compact formula we need one more piece of notation: for $x \in E$, $x = a_1 v_1 + \dots + a_d v_d$, let

$$[x] = \{v_i\colon\ a_i \neq 0\}.$$

Lemma 11. *Let $D \subset \mathbb{R}^d$ be a simplicial cone of dimension d generated by the linearly independent set $V = \{v_1, \dots, v_d\} \subset \mathbb{Z}^d$. Let G be the union of some facets F of D, and set $W = \bigcup_{F \subset G} V \setminus F$. Then*

$$\begin{aligned} H_{D \setminus G}(\boldsymbol{t}) &= \mathscr{H}_V(\boldsymbol{t}) \sum_{x \in E} \boldsymbol{t}^{-x} \boldsymbol{t}^{W \cup [x]} \\ &= \mathscr{H}_V(\boldsymbol{t}) \sum_{x \in E} \boldsymbol{t}^{x} \boldsymbol{t}^{W \setminus [x]}. \end{aligned}$$

Proof. Let $Y \subset V = \{v_1, \dots, v_d\}$. For simplicity of notation we set

$$E(Y) = \operatorname{par}(Y) \cap \mathbb{Z}^d = \bigl\{x \in E\colon\ [x] \subset Y\bigr\}.$$

Moreover, note that since D is simplicial, a face of D is not contained in G if and only if it contains W. Then

$$\begin{aligned} H_{D \setminus G} &= \sum_{Y \supset W} H_{\operatorname{int}(\mathbb{R}_+ Y)}(\boldsymbol{t}) = \sum_{Y \supset W} \sum_{x \in E(Y)} \boldsymbol{t}^{Y} \boldsymbol{t}^{-x} \mathscr{H}_Y(\boldsymbol{t}) = \sum_{x \in E} \boldsymbol{t}^{-x} \sum_{Y \supset W \cup [x]} \boldsymbol{t}^{Y} \mathscr{H}_Y(\boldsymbol{t}) \\ &= \sum_{x \in E} \boldsymbol{t}^{-x} \boldsymbol{t}^{W \cup [x]} \mathscr{H}_{W \cup [x]}(\boldsymbol{t}) \sum_{Z \subset V \setminus (W \cup [x])} \boldsymbol{t}^{Z} \mathscr{H}_Z(\boldsymbol{t}) \\ &= \mathscr{H}_V(\boldsymbol{t}) \sum_{x \in E} \boldsymbol{t}^{-x} \boldsymbol{t}^{W \cup [x]}. \end{aligned}$$

The proof of the second formula is actually simpler. Let L be the free monoid generated by $v_1, \dots, v_d$. Then $D \cap \mathbb{Z}^d = \sum_{x \in E} x + L$. Now one computes $(x + L) \setminus G$, and obtains the result. $\square$

In the present implementation Normaliz uses the first formula in Lemma 11, but only in the case in which there is an integral linear form γ such that the given generators of the cone C have value 1 under γ (this case is called *homogeneous*). Then γ induces a $\mathbb{Z}$-grading on $M = C \cap \mathbb{Z}^d$ in which all generators of all the simplicial cones $C_1, \dots, C_m$ in the triangulation have degree 1, and Lemma 11 specializes to

$$H_{C_i \setminus G}(t) = \frac{1}{(1-t)^d} \sum_{x \in E} t^{|W \cup [x]| - \deg x} = \frac{1}{(1-t)^d} \sum_{x \in E} t^{|W \setminus [x]| + \deg x}. \tag{6.2}$$

Therefore one needs only to count each element $x \in E$ (including 0!) in the right degree to obtain the h-vector of the cone C.

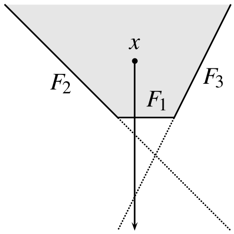

Fig. 4. The line shelling.

The price to be paid for the simple computation of the h-vector is the construction of a shelling. The classical tool for this purpose is a line shelling as introduced by Brugesser and Mani.

First we "lift" the cone $C \subset \mathbb{R}^d$ generated by $v_1, \dots, v_m$ to a cone $C' \subset \mathbb{R}^{d+1}$ by extending the generating elements by positive weights:

$$v_i' = (v_i, w_i) \in \mathbb{Z}^{d+1}, \quad w_i > 0.$$

The *bottom* B of C' is the conical complex formed by all the facets (and their faces) that are "visible from below", more precisely by all the facets F of F' whose corresponding support form $\sigma_F \in (\mathbb{R}^{d+1})^*$ has positive last coordinate. The projection $\mathbb{R}^{d+1} \to \mathbb{R}^d$, $(a_1, \dots, a_{d+1}) \mapsto (a_1, \dots, a_d)$, maps B bijectively onto C, and the images of the facets constitute a conical subdivision of C. We always choose the weights in such a way that the facets in the bottom of C' are simplicial, and therefore we obtain a triangulation of C. (This is the classical construction of regular triangulations; compare [2, 1.F].)

It follows from [15, Theorem 8.1] that this triangulation is shellable, and in order to reduce our conical situation to the polytopal one in [15], one simply works with a suitable polytopal cross-section of C'.

Remark 12. Although it is superfluous, we also keep the "top" of C' simplicial by a suitable choice of weights. The only facets of C' that cannot always be made simplicial are "vertical" ones, namely those parallel to the direction of projection. Each vertical facet of C' corresponds to a (nonsimplicial) facet of C whereas the bottom and top facets correspond to the simplicial cones in triangulations of C. Since such triangulations usually have many more cones than C has support hyperplanes, C' has mainly simplicial facets, and for this reason we have developed the simplicial refinement of Fourier–Motzkin elimination in Section 4.

The proof of [15, Theorem 8.1] tells us how to find a shelling. We choose a point $x \in \operatorname{int}(C')$ such that the ray $x + \mathbb{R}_+ v$, $v = (0, \dots, 0, -1) \in \mathbb{R}^{d+1}$, is intersected at pairwise different points $x + t_F v$ by the linear subspaces $\mathbb{R}F$ where F runs through the facets in the bottom. Then we order the facets by ascending "transition times" t_F. The images of the facets F, ordered in the same way, yield a simplicial shelling of C since the projection preserves the face relation in the complex. The construction of the shelling is illustrated by Fig. 4.

It is not difficult to produce a point x in $\operatorname{int}(C')$, but one may need several attempts to ensure that the transition times are all different. Instead we choose x only once and then replace it by a point infinitely near to x. This trick is known as "simulation of simplicity" in computational geometry (see [9]).

For the next lemma it is convenient to replace the integral support forms σ_F of the bottom faces by their rational multiples $\rho_F = -\sigma_F / \sigma_F(v)$, $v = (0, \dots, 0, -1) \in \mathbb{R}^{d+1}$ as above. These are normed in such a way that $\rho_F(v) = -1$ (and $\rho_F / \sigma_F > 0$).

Lemma 13. *Let the bottom facets of C' be ordered by the following rule: $F < \widetilde{F}$ if $\rho_F(x) < \rho_{\widetilde{F}}(x)$ or $\rho_F(x) = \rho_{\widetilde{F}}(x)$ and ρ_F precedes $\rho_{\widetilde{F}}$ in the lexicographic order on $(\mathbb{R}^{d+1})^*$.*

Then the bottom facets of C' form a shelling in this order.

Proof. Note that there exists a weight vector $w \in \mathbb{R}^{d+1}$ such that ρ_F precedes $\rho_{\widetilde{F}}$ in the lexicographic order if and only $\rho_F(w) < \rho_{\widetilde{F}}(w)$. For sufficiently small $\varepsilon > 0$ our ordering is identical with that obtained from the inequality $\rho_F(x + \varepsilon w) < \rho_{\widetilde{F}}(x + \varepsilon w)$.

The transition time t_F of the ray $(x + \varepsilon w) + \mathbb{R}_+ v$ with the linear subspace spanned by F is given by

$$t_F = -\frac{\rho_F(x + \varepsilon w)}{\rho_F(v)} = \rho_F(x + \varepsilon w),$$

and we have indeed ordered the facets by increasing transition times. $\square$

Remark 14. After the mathematical foundation for the computation of Hilbert functions has been laid in Lemmas 11 and 13, we describe the essential details of the implementation.

(S1) Normaliz computes the support hyperplanes of C—these are needed anyway—and extracts the extreme integral generators from the given set of generators in order to use the smallest possible system of generators for C'.

(S2) The support hyperplanes of C' are computed by Fourier–Motzkin elimination with simplicial refinement as described in Section 4. It is here where the simplicial refinement shows its efficiency since the bottom (and top) facets of C' are kept simplicial by a suitable "dynamic" choice of the weights.
Note that the vertical facets of C', namely those parallel to v, cannot be influenced by the choice of weights. They are determined by the facet structure of C.

(S3) Once the support hyperplanes of C' have been computed, the bottom facets are ordered as described in Lemma 13. Let $C_1 < \cdots < C_m$ be the correspondingly ordered simplicial cones that triangulate C. In order to apply Lemma 11 we have to find the intersections $C_i \cap (C_1 \cup \cdots \cup C_{i-1})$. To this end we do the following: we start with an empty set $\mathscr{F}$, and in step i ($i = 1, \ldots, m$) we insert the facets of C_i into $\mathscr{F}$.
   (i) If a facet is already in $\mathscr{F}$, then it is contained in $C_1 \cup \cdots \cup C_{i-1}$. Since it can never appear again, it is deleted from $\mathscr{F}$.
   (ii) Otherwise it is a "new" facet and is kept in $\mathscr{F}$.

7. Cutting cones by halfspaces

The primal algorithm of Normaliz builds a cone C by starting from 0 and adding the generators $x_1, \ldots, x_n$ successively. The algorithm we want to discuss now (essentially due to Pottier [14]) builds the dual cone C^* successively by staring from 0 and adding generators $\lambda_1, \ldots, \lambda_s$. On the primal side this amounts to cutting out the cone C from $0^* = \mathbb{R}^d$ by successively intersecting the cone reached with the halfspace $H^+_{\lambda_i}$, $i = 1, \ldots, s$, until one arrives at C.

If one wants to compute the Hilbert basis of C via this construction, then one has to understand how to obtain the Hilbert basis of an intersection $D \cap H^+$ from that of the cone D.

Since we start from the full space $\mathbb{R}^d$ and we cannot reach a pointed cone before having cut it with at least d halfspaces, we use the general notion of a Hilbert basis as introduced in Section 2. (In the following we do not assume that C or C^* is a full-dimensional cone.) Of course, in addition to the Hilbert basis B of $M = C \cap \mathbb{Z}^d$, we also need a description of the group $\mathrm{U}(M)$ by a $\mathbb{Z}$-basis.

The halfspace H^+ is given by an integral linear form λ, $H^+ = H^+_\lambda$. In the following the superscript $^+$ denotes intersection with H^+_λ, and the superscript $^-$ denotes intersection with H^-_λ.

There are two cases that must be distinguished:

(a) λ vanishes on $\mathrm{U}(M)$; in this case $\mathrm{U}(M^+) = \mathrm{U}(M^-) = \mathrm{U}(M)$.

(b) λ does not vanish on $\mathrm{U}(M)$; in this case $\mathrm{U}(M^+) = \mathrm{U}(M^-)$ is a proper subgroup of $\mathrm{U}(M)$ such that $\operatorname{rank} \mathrm{U}(M^+) = \operatorname{rank} \mathrm{U}(M) - 1$. Moreover $\mathrm{U}(M)^+$ has a Hilbert basis consisting of a single element h, and then $-h$ constitutes a Hilbert basis of $\mathrm{U}(M)^-$.

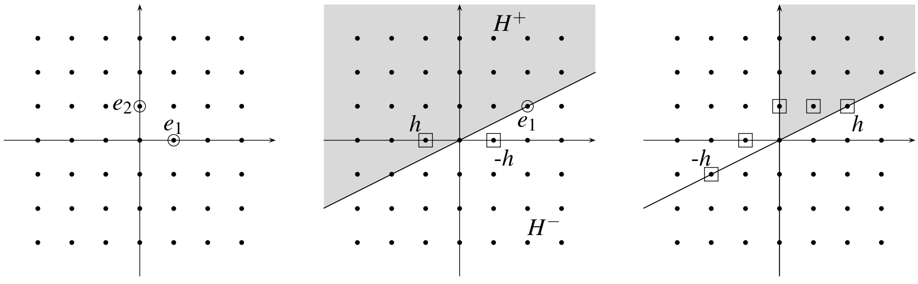

Fig. 5. Successive cuts with halfspaces.

Note that case (a) automatically applies if D is pointed.

The following algorithm computes the Hilbert bases of M^+, M^- and a basis of the group $\mathrm{U}(M^+) = \mathrm{U}(M^-)$, starting from a Hilbert basis B of M and a basis of $\mathrm{U}(M)$.

(D1) Compute a basis of $\mathrm{U}(M^+) = \mathrm{U}(M^-) = \operatorname{Ker}\lambda | \mathrm{U}(M)$. If $\mathrm{U}(M^+) = \mathrm{U}(M)$, then we are in case (a). Otherwise $\operatorname{rank}\mathrm{U}(M^+) = \operatorname{rank}\mathrm{U}(M) - 1$ and we are in case (b).
(D2) In case (b) supplement the basis of $\mathrm{U}(M^+)$ to a basis of $\mathrm{U}(M)$ by an element $h \in \mathrm{U}(M)^+$.
(D3) Set $B_0 = B$.
(D4) In case (b) replace every element $x \in B_0^+$ by $x - ah$ where $a = \lfloor\lambda(x)/\lambda(h)\rfloor$, and every element $x \in B_0^-$ $x - a(-h)$, $a = \lfloor\lambda(-x)/\lambda(h)\rfloor$.
(D5) In case (b) replace B_0 by $B_0 \cup \{h, -h\}$.
(D6) For $i > 0$ set

$$\widetilde{B}_i = B_{i-1} \cup \bigl\{x + y\colon x, y \in B_{i-1},\ \lambda(x) > 0,\ \lambda(y) < 0,\ x + y \neq 0\bigr\}.$$

(D7) Replace $\widetilde{B}_i^+$ by its auto-reduction B_i^+ in D^+, and $\widetilde{B}_i^-$ by its auto-reduction B_i^- in D^-, and let $B_i = B_i^+ \cup B_i^-$.
(D8) If $B_i = B_{i-1}$, then we are done, B_{i-1}^+ is a Hilbert basis of D^+, and B_{i-1}^- is a Hilbert basis of D^-.

The construction is illustrated by Fig. 5; base elements of the unit groups have been marked by a circle, Hilbert basis elements by a square.

We have to prove the claim contained in (D8), and we state it as a lemma.

Lemma 15. *There exists an $i \geqslant 1$ such that $B_i = B_{i-1}$, and in this case $B_i^+ = \mathrm{Hilb}(M^+)$, $B_i^- = \mathrm{Hilb}(M^-)$.*

Proof. Let $B_\infty = \bigcup_{i=0}^\infty B_i$. We will show that B_∞^+ generates M^+ modulo $\mathrm{U}(M^+)$ and B_∞^- does the same for M^-. In other words, we claim that for every $x \in M^+$ there exist $u_1, \dots, u_r \in B_\infty^+$ such that

$$x - (u_1 + \cdots + u_r) \in \mathrm{U}\bigl(M^+\bigr), \tag{7.1}$$

and the corresponding statement holds for B_∞^- and M^-.

Suppose this claim has been proved. Then B_∞^+ contains a Hilbert basis of M^+ since it is a system of generators modulo $\mathrm{U}(M^+)$. Since the Hilbert basis contains only irreducible elements (an irreducible element will pass step (D7) above) and is finite, there must be an i for which B_{i-1}^+ contains the Hilbert basis (in fact, equals it). Then $B_i^+ = B_{i-1}^+$. Increasing i if necessary, we also have $B_i^- = B_{i-1}^-$, and then $B_i = B_{i-1}$. Conversely, if $B_i = B_{i-1}$, then $B_{i-1}^+ = \mathrm{Hilb}(M^+)$ and $B_{i-1}^- = \mathrm{Hilb}(M^-)$.

We (have to) prove the crucial claim simultaneously for M^+ and M^-, considering the more complicated case (b). The proof for case (a) is obtained if one omits all those arguments that refer to h.

We use induction on $\operatorname{tdeg} x$, the total degree with respect to M. We can assume that $x \in M^+$ since the argument for $x \in M^-$ is analogous. If $\operatorname{tdeg} x = 0$, we have $x \in \mathrm{U}(M)$. But then $x - ah \in \mathrm{U}(M^+)$, since h is a Hilbert basis of $\mathrm{U}(M)^+$ modulo its group $\mathrm{U}(M^+)$ of invertible elements. Moreover, $h \in B_\infty^+$ by construction.

Suppose that $\operatorname{tdeg} x > 0$, and note that x has a representation

$$x \equiv (u_1 + \cdots + u_r) + (v_1 + \cdots + v_s) + (w_1 + \cdots + w_t) \mod \mathrm{U}\bigl(M^+\bigr) \tag{7.2}$$

modulo $\mathrm{U}(M^+)$ in which $u_j, v_k, w_l \in B_\infty$ and $\lambda(u_j) > 0$, $\lambda(v_k) = 0$ and $\lambda(w_l) < 0$. In fact, since B_∞ contains a Hilbert basis of M, we can find such a representation modulo $\mathrm{U}(M)$, and adding h or $-h$ sufficiently often, we end up in $\mathrm{U}(M^+)$.

Among all the representations (7.2) we choose an *optimal* one, namely one for which $\lambda(u_1 + \cdots + u_r)$ is minimal. If we can show that $t = 0$ for this choice, then we are done. Note that only one of h or $-h$ can appear in an optimal representation; otherwise canceling h against $-h$ would improve it.

Clearly, if $t > 0$, then $r > 0$ as well, since otherwise $\lambda(x) \geqslant 0$ is impossible. Consider the representation

$$x \equiv (u_1 + w_1) + (u_2 + \cdots + u_r) + (v_1 + \cdots + v_s) + (w_2 + \cdots + w_t) \mod \mathrm{U}\bigl(M^+\bigr) \tag{7.3}$$

modulo $\mathrm{U}(M^+)$.

If $u_1 + w_1$ belongs to B_∞ we are done, since $\lambda(u_1 + w_1) < \lambda(u_1)$, regardless of the sign of $\lambda(u_1 + w_1)$.

Otherwise $u_1 + w_1$ is reducible in step (D7). Assume $u_1 + w_1 \in M^+$ (an analogous argument can be given if $u_1 + w_1 \in M^-$). Then there exists $y \in B_\infty^+$ such that $(u_1 + w_1) - y \in M^+$. Note that $y = h$ is impossible: by construction, all elements z of B_∞ different from h and $-h$ have $|\lambda(z)| < \lambda(h)$, so $\lambda(u_1 + w_1) < \lambda(u_1) < \lambda(h)$ and $(u_1 + w_1) - h \notin M^+$. But all elements of B_∞ different from h and $-h$ do not belong to $\mathrm{U}(M)$ and therefore have positive total degree in M. Then $\operatorname{tdeg}(u_1 + w_1 - y) < \operatorname{tdeg}(u_1 + w_1)$. Thus we can apply induction to $(u_1 + w_1) - y$, representing it modulo $\mathrm{U}(M^+) = \mathrm{U}(M^-)$ by elements from B_∞^+. Since $y \in B_\infty^+$, we obtain a representation for $u_1 + w_1$. Substituting this representation into (7.3) again yields an improvement. This is a contradiction to the choice of (7.2), and we are done. □

While the description of the algorithm given above is very close to the implementation in Normaliz, we would like to mention some further details.

Remark 16. (a) It is clear that in the formation of $\widetilde{B}_i$ in step (D6) one should avoid the sums $x + y$ that have already been formed in an earlier "generation".

(b) When a sum $x + y$ has been formed, it is immediately tested against reducibility by B_{i-1}^+ or B_{i-1}^-, respectively. The elements that survive are collected, and the remaining reduction steps are applied after this collection.

This "generation driven" procedure has the advantage that we can apply the rather efficient reduction strategy of Section 3.

(c) Sometimes an element is reduced by one that is created in a later generation. However, this happens rarely.

(d) When a sum $z = x + y$ is formed and belongs to $\widetilde{B}_i^+$, then we store $\lambda(x)$ with z (analogously if $x + y \in M^-$). Suppose that z survives the reduction. Then it is not necessary to form sums $z + w$ with $\lambda(w) < -\lambda(x)$ since we would have $x + w, z + w \in M^-$, and $x + w$ clearly reduces $z + w$. This trick diminishes the number of sums to be formed in higher generations considerably, but does not help in the formation of B_1, which is usually (but not always) the most time consuming generation.

(e) Normaliz uses a heuristic rule to determine the order in which the hyperplanes are inserted into the algorithm. It is evidently favorable to keep the sets B_i small as long as possible.

(f) In a future version of Normaliz we will also try a hybrid approach in which the algorithm for the local Hilbert bases is chosen dynamically.

We conclude the article by a comparison of the primal algorithm of Normaliz and the algorithm of this section.

Remark 17. (a) Normaliz has two input modes in which the cone C is specified by inequalities (and equations). In these cases the user can choose whether to apply the primal algorithm (first computing a system of generators of C) or the dual algorithm described in this section. It is not easy to decide which of the two algorithms will perform better for a given C. The bottleneck of the primal algorithm is certainly the computation of a full triangulation (if it is done). The size of the triangulation is mainly determined by the number of support hyperplanes of the subcones of C through which the computation passes. However, if it can be found, the (partial) triangulation itself carries a large amount of information, and the subsequent steps profit from it.

We illustrate these performance of the primal algorithm by two examples, one for which it is very fast, and another one that it cannot solve.

- (i) The example `small` from the Normaliz distribution is defined by a 5-dimensional lattice polytope with 190 vertices, 32 support hyperplanes, and 34,591 lattice points. Its normalized volume is 2,276,921, and the triangulation contains 1593 simplicial cones (if computed in the mode "normal"). About 230,000 vectors survive the local reduction, and are sent into global reduction, leading to the Hilbert basis with 34,591 vectors. (The number of candidates for global reduction caries considerably with the triangulation; those derived from shellings seem to behave worse in this respect.) Run time with Normaliz 2.2 (the currently public version) on our SUN Fire X4450 is 8 sec (and 19 sec if the h-vector is computed).
- (ii) The example `5x5` from the Normaliz distribution describes the cone of 5×5 "magic squares" [1], i.e., 5×5 matrices with nonnegative entries and constant row, column and diagonal sums. The cone of dimension 15 has 1940 extreme rays and 25 support hyperplanes. The subcone generated by the first 57 extreme rays (in the order Normaliz finds them) has already 30,290 support hyperplanes. After 104 extreme rays we reached 56,347 support hyperplanes (and we stopped the program).

(b) The main obstruction in the application of the dual algorithm is the potentially extremely large number of vectors it has to generate. Even if the Hilbert basis of the final cone C is small, the Hilbert bases of the overcones of C through which the algorithm passes may be extremely large, or one has to compute with medium size Hilbert bases in many successive overcones. Some data on the behavior of the algorithm on the two examples from (a)—now (i) is hard, and (ii) is easy:

- (i) For the lattice polytope the dual algorithm (staring from the support hyperplanes) needs 3540 sec. One of the intermediate Hilbert bases has cardinality 145,098. Therefore the number of elements of $\widetilde{B}_1$ at the insertion of the next hyperplane can safely be estimated by 10^9.
- (ii) After 20 hyperplanes have been inserted, the size of the Hilbert basis of the cone reached is 228, and the values for the subsequent cones are 979, 1836, 2810, 3247, and finally 4828. Computation time is 2 sec.

(c) If the Hilbert basis of $C = \mathbb{R}_+ x_1 + \cdots + x_n$ is to be computed, the primal algorithm builds an ascending chain

$$0 = C_0 \subset C_1 \subset \cdots \subset C_n = C,$$

with a corresponding decreasing chain

$$\left(\mathbb{R}^d\right)^* = C_0^* \supset C_1^* \supset \cdots \supset C_n^* = C^*.$$

For the cone $D = \bigcap_{j=1}^s H_{\lambda_i}^+$ the dual algorithm proceeds in exactly the opposite way, building an increasing chain

$$0 = D_0^* \subset D_1^* \subset \cdots \subset D_s^* = D^*,$$

of dual cones, with a corresponding decreasing chain

$$\mathbb{R}^d = D_0 \supset D_1 \supset \cdots \supset D_s = D.$$

In both cases, the complexity is determined by the *decreasing* chains of overcones of C^* and D, respectively. These overcones are hard to control only by the internal data of C or D^*.

Of course, if $n = 1940$ and $s = 25$ as in example (ii), then the choice is easy, but example (i) with $n = 190$ and $s = 32$ illustrates that the sole comparison of n and s does in general not help to pick the better algorithm.

We conclude by presenting the following table which contains experimental test data we have obtained for computing the Hilbert basis with Normaliz version 2.2, as well as data obtained from our tests with 4ti2 version 1.3.2. The system 4ti2 [10,11] contains a somewhat different implementation of the dual algorithm. The table shows that our version is certainly comparable in performance.

Name	Input	dim	#gen	#supp	#HB	t primal	t dual	t 4ti2
cut.in	supp	5	83	25	4398	0.15	3.4	620
small.in	gen	6	190	32	34,591	8	3540	3230
medium.in	gen	17	40	3753	217	11	∞	∞
4x4.in	equ	8	20	34	20	0.002	0.001	0.01
5x5.in	equ	15	1940	47	4828	∞	2	2.4
6x6.in	equ	24	97,548	62	522,347	∞	87,600	345,600

The first column refers to the name of the input file in the Normaliz distribution. The second column describes the type of input, namely *gen*erators, *supp*ort hyperplanes, or system of *equ*ations. In the latter case the cone is the intersection of the solution space with the nonnegative orthant. The third column contains the dimension of the cone, and the following three list the number of its generators, support hyperplanes and Hilbert basis elements. The last three columns contain computation times for Normaliz primal, Normaliz dual and 4ti2 (measured in seconds). For the application of Normaliz dual or 4ti2 to input of type "gen", we first computed the dual cone separately (or extracted it from the output of the primal algorithm). Normaliz primal, when applied to any type of input, does the necessary dualization itself.

We thank Christof Söger for measuring the computation times.

References

[1] M. Ahmed, J. De Loera, R. Hemmecke, Polyhedral cones of magic cubes and squares, in: B. Aronov, et al. (Eds.), Discrete and Computational Geometry. The Goodman–Pollack Festschrift, Springer, 2003, pp. 25–41.
[2] W. Bruns, J. Gubeladze, Polytopes, Rings, and K-Theory, Springer, 2009.
[3] W. Bruns, J. Herzog, Cohen–Macaulay Rings, rev. ed., Cambridge University Press, 1998.
[4] W. Bruns, R. Hemmecke, B. Ichim, M. Köppe, C. Söger, Challenging computations of Hilbert bases of cones associated with algebraic statistics, preprint http://arxiv.org/1001.4145.
[5] W. Bruns, B. Ichim, Normaliz. Computing normalizations of affine semigroups, with contributions by C. Söger, available from http://www.math.uos.de/normaliz.
[6] W. Bruns, B. Ichim, Introduction to Normaliz 2.2, in: D. Breaz, et al. (Eds.), Proceedings of ICTAMI, Acta Univ. Apulensis (2009) 113–132.
[7] W. Bruns, G. Kämpf, A Macaulay 2 interface for Normaliz, preprint http://arxiv.org/abs/0908.1308.
[8] W. Bruns, R. Koch, Computing the integral closure of an affine semigroup, Univ. Iagel. Acta Math. 39 (2001) 59–70.
[9] H. Edelsbrunner, Algorithms in Combinatorial Geometry, Springer, 1987.
[10] R. Hemmecke, On the computation of Hilbert bases of cones, in: A.M. Cohen, et al. (Eds.), Mathematical Software, Proceedings of the 1st International Congress, World Scientific, 2002, pp. 307–317.
[11] R. Hemmecke, R. Hemmecke, P. Malkin, 4ti2. Version 1.3.2. Computation of Hilbert bases, Graver bases, toric Gröbner bases, and more, available from http://www.4ti2.de.
[12] M. Joswig, Beneath-and-beyond revisited, in: M. Joswig, et al. (Eds.), Algebra, Geometry, and Software Systems, Springer, 2003, pp. 1–21.

[13] M. Joswig, B. Müller, A. Paffenholz, Polymake and lattice polytopes, in: C. Krattenthaler, et al. (Eds.), DMTCS proc. AK, Proceedings of FPSAC, 2009, pp. 491–502.
[14] L. Pottier, The Euclide Algorithm in Dimension *n*, Research report, ISSAC 96, ACM Press, 1996.
[15] G.M. Ziegler, Lectures on Polytopes, Springer, 1994.