Traversing Symmetric Polyhedral Fans

Anders Nedergaard Jensen*

Mathematisches Institut, Georg-August-Universität Göttingen, Bunsenstraße 3-5, D-37073 Göttingen, Germany

Abstract. We propose an algorithm for computing the facets of a pure connected polyhedral fan up to symmetry. The fan is represented by an oracle. With suitable implementations of the oracle the same algorithm can be used for computing secondary fans, Gröbner fans, Tropical varieties and Minkowski sums up to symmetry. The algorithm has been implemented in the software Gfan.

Keywords: Polyhedral fans, tropical geometry, algorithms, symmetry

1 Introduction

Polyhedral fans arise naturally in convex geometry, with the prime example being secondary fans whose cones index all combinatorial types of polyhedra with a fixed set of normals. In algebraic geometry they give rise to toric varieties and play the central role in the evolving field of tropical geometry. This paper is concerned with the problem of computing polyhedral fans up to symmetry.

Exploiting symmetries in computational geometry is not a new idea. Indeed the method we present here specializes to the well-known *adjacency decomposition method* when the fan to be traversed is full-dimensional; see [3]. In the case of secondary fans, our work can be viewed as a refinement of [11].

We use the following example as a motivation for our approach.

Example 1. Consider the family of curves in $\mathbb{C}^2$ each defined by a polynomial

$$f = a + bx + cx^2 + dx^2y + ex^2y^2 + gxy^2 + hy^2 + iy + jxy \in \mathbb{C}[x, y].$$

A point on a curve is called a *cusp* if

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} = 0 \text{ and } \frac{\partial^2 f}{\partial x \partial y}\frac{\partial^2 f}{\partial y \partial x} - \frac{\partial^2 f}{\partial x \partial x}\frac{\partial^2 f}{\partial y \partial y} = 0$$

in that point. Eliminating variables x and y we get an ideal I_{cusp} defining the subfamily of curves with a cusp. We will be interested in the tropical variety $T(I_{\text{cusp}})$ which is a polyhedral fan. The reduced Gröbner basis for I_{cusp} with respect to the term order given by the point in the support of $T(I_{\text{cusp}})$

$$(304, -158, -152, -206, 388, -248, -146, -128, 346) \in \mathbb{R}^9, \qquad (1)$$

tie-broken reverse lexicographically, has 18608 terms in 23 polynomials.

* Supported by the German Research Foundation (Deutsche Forschungsgemeinschaft (DFG)) through the Institutional Strategy of the University of Göttingen.

In [2] it was suggested that the Gröbner cones are represented by Gröbner bases when computing tropical varieties. The example shows that for practical purposes it is important not to store all these bases in memory. The algorithm we present keeps only one such algebraic representation stored at a time.

For clarity, the fan will be given to us by an oracle. Our main contributions are a description of a symmetry exploiting traversal algorithm with a minimal number of oracle calls, a practical method for checking orbit membership of cones and finally a description of an oracle implementation for the restriction of a Gröbner fan to a lower-dimensional polyhedral cone.

We give a few more details on the implementation in the software Gfan [10] and end the paper by computing the fan in Example 1 using this software.

2 Definitions and Notation

By a *polyhedral cone* $C \subseteq \mathbb{R}^n$ we mean a finite intersection of closed halfspaces in $\mathbb{R}^n$, or, equivalently, the non-negative span $\mathrm{cone}(v_1, \ldots, v_m)$ of a collection of vectors $\{v_1, \ldots, v_m\} \subseteq \mathbb{R}^n$. We use $\mathrm{rel\,int}(C)$ to denote the *relative interior* of C. The inclusion largest linear subspace contained in C is called the *lineality space* of C. It equals $C \cap -C$. For an $\omega \in \mathbb{R}^n$ we let $\mathrm{face}_\omega(C)$ denote the *face* of C at which $\langle \omega, \cdot \rangle$ is maximized and use the same notation for polytopes. A finite collection $\mathcal{F}$ of cones is called a *polyhedral fan* if

- $C \in \mathcal{F}$ implies that every face of C is in $\mathcal{F}$, and
- $C, C' \in \mathcal{F}$ implies that $C \cap C'$ is a face of C.

In particular, the cones in a fan must all have the same common lineality space. An example of a fan is the set of faces $\mathrm{faces}(C)$ of a cone C. The *common refinement* $A \wedge B := \{a \cap b : (a, b) \in A \times B\}$ of two fans A and B is a fan. The *f-vector* of $\mathcal{F}$ lists the number of cones of each dimension, starting with a 1 for the lineality space. The *support* $\mathrm{supp}(\mathcal{F})$ of a fan $\mathcal{F}$ is the union of its cones. We will use the untraditional word *ray* to denote a cone with exactly two faces (the cone and its lineality space). For a rational ray C the intersection $(C \cap -C)^\perp \cap C$ is a one-dimensional half-line which has a unique first non-zero lattice point in $\mathbb{Z}^n$. We call this the *primitive vector* of C and denote it $\mathrm{prim}(C)$. Let the *link* of a cone C at a point $v \in C$ be

$$\mathrm{link}_v(C) = \{u \in \mathbb{R}^n : \exists \delta \in \mathbb{R}_{>0} : \forall \varepsilon \in (0, \delta) : v + \varepsilon u \in C\},$$

and define the link of a fan $\mathcal{F}$ at a point v in the support of $\mathcal{F}$ to be the fan

$$\mathrm{link}_v(\mathcal{F}) = \{\mathrm{link}_v(C) | v \in C \in \mathcal{F}\}.$$

Since any two points in the relative interior of a cone $R \in \mathcal{F}$ will give the same link, we will also denote the link $\mathrm{link}_R(\mathcal{F})$. In the special case where R is a *facet* of C, meaning that the dimension of R is one smaller than the dimension of C, the link $\mathrm{link}_R(C)$ is a ray and we also denote it by $\mathrm{ray}(R, C)$.

An inclusion maximal cone in a fan is called a *facet*. We shall be mainly interested in *pure* fans which are fans whose facets all have the same dimension d. A cone of dimension $d-1$ in such a fan is called a *ridge*. A *ridge path* in $\mathcal{F}$ is a sequence of facets $F_1, \ldots, F_s$ such that $F_i \cap F_{i+1}$ is a ridge. A pure fan is *connected in codimension one* if any two facets are connected by a ridge path. The symmetric group S_n acts on $\mathbb{R}^n$ by permuting coordinates. This action extents to cones and fans in $\mathbb{R}^n$. A subgroup $G \subseteq S_n$ is said to be a *symmetry group* of a fan $\mathcal{F}$ if it is contained in the stabilizer of $\mathcal{F}$.

2.1 Gröbner fans and tropical varieties

We consider the polynomial ring $k[x_1, \ldots, x_n]$ over a field k. For a vector $\omega \in \mathbb{R}^n$, the *initial form* $\text{in}_\omega(f)$ of a polynomial $\sum_i c_i x^{a_i}$ with $c_i \in k \setminus \{0\}$ and $a_i \in \mathbb{N}^n$ is defined as the sum of all terms $c_i x^{a_i}$ such that $\langle \omega, a_i \rangle$ is maximal. We define the *initial ideal* of I as $\text{in}_\omega(I) := \langle \text{in}_\omega(f) : f \in I \rangle$. Now, fix an ideal I. Two vectors $u, v \in \mathbb{R}^n$ are equivalent if $\text{in}_u(I) = \text{in}_v(I)$. The closure of an equivalence class containing $v \in \mathbb{R}^n_{>0}$ is a polyhedral cone $\mathcal{C}_v(I)$ and the collection of all cones and their faces is the *Gröbner fan* $\Sigma(I)$ of I. The maximal cones of $\Sigma(I)$ are in bijection with the *marked reduced Gröbner bases* of I. They are reduced Gröbner bases where the initial term of each polynomial has been distinguished – it has been marked. Given a term order $\prec$ we use the notation $\mathcal{G}_\prec(I)$ for the marked reduced Gröbner basis with respect to $\prec$ and $\mathcal{C}_\prec(I)$ for its cone. If I is homogeneous, then the fan is complete and we define the *tropical variety* $T(I)$ of I to be the following subfan of the Gröbner fan $\Sigma(I)$: $T(I) := \{\mathcal{C}_v(I) : \text{in}_v(I) \text{ contains no monomial}\}$. See [2] and [8] for details.

3 The traversal algorithm

In this section we present an algorithm for traversing the maximal cones of a pure d-dimensional, codimension-one-connected fan $\mathcal{F}$. Explaining the algorithm in great detail makes it easy for us to be precise in Section 3.1 where we will modify the algorithm to exploit symmetry. The fan $\mathcal{F}$ is known to the algorithm only through an oracle. The oracle allows two main operations:

- Given a maximal cone $C \in \mathcal{F}$ and a facet R of C we may ask for $\text{link}_R(\mathcal{F})$. The link is a list of rays $O_C^{link}(R)$.
- Given a maximal cone C, a facet R of C and a ray $v \in \text{link}_R(\mathcal{F})$ we may ask for the cone $O_C^{change}(R, v)$ in $\mathcal{F}$ having link v at R.

The subscript in our oracle notation needs more explanation. We do not allow oracle calls in arbitrary order but think of the oracle as having an internal state being a facet $C \in \mathcal{F}$ and additional information. We may only ask for $O_C^{link}(R)$ and $O_C^{change}(R, v)$ when the oracle is in state C. The oracle call $O_C^{change}(R, v)$ changes the state to C', where C' is the returned maximal cone giving rise to v in the link. In addition to the above calls, we are allowed to ask the oracle which cone it is in; $O_C^{cone}()$ will return C, but it will not reveal the complete state.

Furthermore, we will assume that the oracle is in some state at the beginning of the traversal.

The following example illustrates admissible oracle call sequences. It will be a Gröbner fan to emphasize that the state may consists of non-geometric data.

Example 2. Consider the ideal $I := \langle a^2 + bc, b^2 + ac, c^2 + ab \rangle \subseteq \mathbb{Q}[a, b, c]$. A computation reveals that $\Sigma(I)$ is a three-dimensional fan in $\mathbb{R}^3$ with f-vector $(1, 9, 9)$ and a 1-dimensional lineality space. The 9 rays of the fan are generated by the lineality space and one of the vectors

$$(1, -1, 0), (0, -1, 1), (-1, -1, 2), (-1, 0, 1), (-1, 1, 0),$$

$$(-1, 2, -1), (0, 1, -1), (1, 0, -1), (2, -1, -1),$$

which are ordered cyclically. By Gröbner basis theory the nine maximal cones are in bijection to the nine reduced Gröbner bases of I. One of these is

$$\{\underline{c^2} + ab, \underline{bc} + a^2, \underline{b^2} + ac, \underline{a^2c}, \underline{a^2b}, \underline{a^4}\}$$

corresponding to the maximal cone $C_1 := \text{cone}(\pm(1, 1, 1), (-1, 0, 1), (-1, 1, 0))$. Let R_1 be the ridge $\text{cone}(\pm(1, 1, 1), (-1, 0, 1))$. An oracle representing the Gröbner fan would have

$$\begin{aligned} O^{link}_{C_1}(R_1) = \{&\text{cone}(\pm(1, 1, 1), \pm(-1, 0, 1), (-1, 2, -1)), \\ &\text{cone}(\pm(1, 1, 1), \pm(-1, 0, 1), (1, -2, 1))\}. \end{aligned}$$

Later we shall be less strict and think of this as just a set of two vectors, but because of scaling and the non-trivial lineality space there are several possibilities for choosing these representatives.

Let $v := \text{cone}(\pm(1, 1, 1), \pm(-1, 0, 1), (1, -2, 1))$. Making the call

$$C_2 := O^{change}_{C_1}(R_1, v) = \text{cone}(\pm(1, 1, 1), (-1, -1, 2), (-1, 0, 1))$$

changes the state to C_2. Now the calls $O^{link}_{C_1}(R_1)$ and $O^{change}_{C_1}(R_1, v)$ are illegal, while $O^{link}_{C_2}(R_1)$ and $O^{change}_{C_2}(R_1, -v)$ are legal. Applying a total of nine O^{change} calls we can return to state C_1.

Since the hidden state information can be huge, see Example 1, our calling conventions for the oracle have been designed so that only one state is stored, keeping memory consumption as low as possible. We note that reconstructing the hidden state information from a polyhedral cone can be quite complicated. Indeed, the Gröbner walk [5] speeds up the process of computing a Gröbner basis with respect to a prescribed term order by making a sequence of local changes.

We cannot always, as in Example 2, think of the facets of $\mathcal{F}$ as being vertices of a graph with the ridges being edges connecting them, since a ridge may connect more than two facets if the fan is not full-dimensional. Rather we should think of a hypergraph, in which the hyperedges connect many vertices. Traversing a hypergraph by an exhaustive search is not more complicated than traversing a graph. In fact our problem translates into traversing the bipartite graph $G_{\mathcal{F}}$ with the right hand side being the facets, the left hand side being the ridges, and two cones being connected if one is contained in the other. Having $\mathcal{F}$ connected in codimension one is equivalent to $G_{\mathcal{F}}$ being connected.

We recall a basic graph traversal algorithm for connected graphs:

Algorithm 1
Input: *A connected graph* $G = (V, E)$*, a vertex* $v \in V$*.*
Output: *All vertices* V *of* G*.*

– $(A, B, D) := (\{v\}, \emptyset, \emptyset)$;
– *while*($A \neq \emptyset$)
  • *Choose* $u \in A$;
  • $A := A \setminus \{u\}$;
  • $B := B \ominus \{\{a, u\} : \{a, u\} \in E\}$;
  • $D := D \cup \{u\}$;
  • $A := A \cup \{a : \{a, u\} \in B\}$;
– *output* D;

Here $S \ominus T$ denotes the symmetric difference $(S \cup T) \setminus (S \cap T)$ of two sets S and T. An invariant for the algorithm is that after each step the edge set B is the boundary of the vertex set D. At the end $D = V$ and B and A are empty.

We will use a depth-first approach to traverse the bipartite graph $G_{\mathcal{F}}$. This means that the set A will work as a last-in-first-out stack. Equivalently, we may present the above algorithm as two mutually recursive procedures, with left and right hand side nodes being treated differently. Thus, in the algorithm below, the set A is stored implicitly on the recursion stack while facets are written to the output rather than stored in the set D. The set B will no longer be a collection of sets of cones from $\mathcal{F}$, but rather consist of pairs of the form (R, v), where R is a ridge of $\mathcal{F}$ and v is a ray in $\text{link}_R(\mathcal{F})$.

Algorithm 2
Input: *An oracle* O *in state* C_0 *representing a codimension* 1 *connected fan* $\mathcal{F}$*.*
Output: *All facets of* $\mathcal{F}$*.*

– $B := \emptyset$;
– *Call* **EnumerateFacet(**C_0**)** *below;*

EnumerateFacet(C**)**

– *Output* C;
– *Compute the facets of* C;
– $T := \{(R, \text{ray}(R, C)) : R \text{ is a facet of } C\}$;
– $B := B \ominus T$;
– *For every pair* $(R, v) \in T$
  • *If* $(R, v) \in B$ *then call* **EnumerateRidge(**R, C**)**;

EnumerateRidge(R, C)

- $L := O_C^{link}(R)$;
- $T := \{(R, v) : v \in L\}$;
- $B := B \ominus T$;
- $v' := \mathrm{ray}(R, C)$;
- *For pair* $(R, v) \in T$
  - *If* $(R, v) \in B$ *then*
    - $C := O_C^{change}(R, v)$
    - *Call* **EnumerateFacet**(C);
    - $C := O_C^{change}(R, v')$

Proof. The algorithm is a direct translation of Algorithm 1 as explained above. We note that at any time the oracle is in state C and that after calling EnumerateRidge, EnumerateFacet sets C to the original second argument value by calling the oracle. This shows that the sequence of oracle calls is valid.

We measure the efficiency of a traversal strategy by the maximal number of $O_C^{link}(R)$ and $O_C^{change}(R, v)$ oracle calls needed as functions of the number of ridges and facets in the fan, respectively. An enumeration strategy is considered optimal if these functions are minimal among all strategies.

Proposition 1. *Let r be the number of ridges in $\mathcal{F}$ and f the number of facets. Algorithm 2 makes r oracle calls of type $O_C^{link}(R)$ which is optimal. It makes* $2(f-1)$ *oracle calls of type $O_C^{change}(R, v)$. By postponing the last oracle call $C := O_C^{change}(R, v')$ until absolutely needed, Algorithm 2 makes at most* $\max(2f-3, 0)$ *oracle calls of type $O_C^{change}(C, v)$. This is optimal.*

Proof. The result follows from the fact that EnumerateRidge is called once for every ridge, and that it does two $O_C^{change}(R, v)$ calls for every facet except C_0. The number of $O_C^{link}(R)$ calls is optimal, since every ridge must be investigated. We never have to bring the oracle back to the initial state at the end. This reduces the number of oracle calls by at least one. To see that this is optimal we consider a worst case scenario of a pure connected fan with f facets on a ridge path and $f+1$ ridges. In an unlucky case the oracle starts close to one end of the fan, moves to the other end and is forced to go back to finish the job. This gives $2f-3$ calls.

Whether $f-1$ is the optimal number of $O_C^{change}(C, v)$ calls *for a particular graph $G_{\mathcal{F}}$* depends on the topology of $G_{\mathcal{F}}$ and is related to the Hamiltonian path problem. We note that $f-1$ is optimal if we relax the oracle call order restriction, but that this would increase memory usage in practice.

For practical implementations of Algorithm 2 it can be an advantage to represent the elements in B as pairs of vectors. For example, (R, v) can be represented by a pair of deterministically computed points in $\mathrm{rel\ int}(R)$ and $\mathrm{rel\ int}(v)$. We will return to the choice of these vectors in the next section.

3.1 Exploiting symmetry

In addition to the oracle, Algorithm 2 could be changed to take a subgroup G of symmetries under which $\mathcal{F}$ is known to be invariant as input. Our goal would then be to find all orbits of maximal cones $\mathcal{F}$ under this group action. We shall restrict ourselves to symmetries which are coordinate permutations and assume that $G \subseteq S_n$. However, this restriction will only be important when we define $p(C)$ later in this section (where $\mathbb{Z}^n$ must be preserved) and for Algorithm 3.

First we define what we mean by a *canonical representative* for the orbit of a pair of cones (R, v), where R is a ridge of $\mathcal{F}$ and $v \in \mathrm{link}_R(\mathcal{F})$. Fix a total order $\prec$ on the set K of polyhedral cones in $\mathbb{R}^n$. We define $\mathrm{CanRep}(R, v)$ to be the smallest element in $\{(\sigma(R), \sigma(r)) : \sigma \in G\}$, with the ordering being the lexicographic order on $K \times K$ with each K ordered by $\prec$.

We now explain how to change Algorithm 2 to compute just one facet (and one ridge) of each orbit. To be precise we will avoid calls EnumerateRidge(R,C) if the procedure has already been called for another ridge in the orbit of R. Similarly, we avoid calls EnumerateFacet(C) and the two surrounding oracle calls if the procedure has already been called for another facet in the orbit of C. Equivalently, we traverse the bipartite *quotient graph* $\overline{G_{\mathcal{F}}}$, where vertices are identified if they are in the same orbit and multi-edges are regarded as single edges.

Three kinds of changes are required:

- In both procedures of Algorithm 2 we let T consist of the canonical representatives of the orbits of the pairs with respect to G rather than the pairs themselves. This may make T smaller since only one element from each orbit can be in T.
- At the two places where we check for containment of (R, v) in B, we should instead check for containment of $\mathrm{CanRep}(R, v)$.
- When we recursively call EnumerateRidge after having checked that B contains $\mathrm{CanRep}(R, v)$, we need to recover (one of) the original facet(s) of C giving rise to R. That is we must find the σ we applied to get R in T. We then use $\sigma^{-1}(R)$ when calling EnumerateRidge. Similarly, when we in EnumerateRidge have verified that $\mathrm{CanRep}(R, v)$ is in B, we must find the (or one) $v \in L$ giving rise to the $\mathrm{CanRep}(R, v)$ in T. We will use this v when calling the oracle.

In the above description we do operations on polyhedral cones when handling symmetries, but this is not convenient in practice. Rather, for a cone C we wish to define a canonical, symmetry invariant relative interior point $p(C)$. In particular, we must have $\sigma(p(C)) = p(\sigma(C))$ for every $\sigma \in G$ and cone $C \subseteq \mathbb{R}^n$. Checking if two cones of $\mathcal{F}$ are in the same orbit can be done by checking that their points are in the same orbit. Even better, for a pair of ridge-facet incidences represented by (R, v) and (R', v') we can check if they are the same up to symmetry by checking if $(p(R), p(v))$ and $(p(R), p(v))$ are the same up to symmetry.

We notice that the vector $p(C) := \sum \mathrm{prim}(r)$, where r runs over all rays of C, satisfies the above properties. However, this definition has the disadvantage that computing it requires knowing the extreme rays of C. Often C is simplicial and this is not a problem, but in general an H-to-V conversion is needed. Alternatively, we may define $p(C)$ using *analytic centers* of polytopes, which can be computed in polynomial time by numerical methods. We have no practical experience with this approach.

3.2 Symmetry algorithms

Complexity-wise, deciding if two vectors in $\mathbb{Z}^n$ are the same up to the action of a group $G \subseteq S_n$, specified by its generators, is as hard as the graph isomorphism problem of deciding if two graphs are the same up to permutation of their vertices. Indeed asking if the edge-vertex incidence matrices of the two graphs are the same up to row and column interchanges answers the question. The graph isomorphism problem is not known to be in P, the class of polynomial time solvable problems, and therefore we cannot expect the canonical representative computation to have polynomial time complexity. We discuss how to solve the problem in practice.

We will not address the problem of computing generators for our group G but rather suppose that they are given. Each generator can be represented by a permutation of the vector $(1, 2, \ldots, n)$. We start by precomputing all elements of G and store them in a *prefix tree* (or a *trie*). A prefix tree has an integer at each node (except the root), and it represents all vectors of integers we get by going from the root to a leaf, picking up integers from the nodes we pass through. Thus we will use a tree of depth n. We are seeking an algorithm with the following specification.

Algorithm 3
Input: *A subgroup $G \subseteq S_n$ stored in a prefix tree and vectors $R, v \in \mathbb{Z}^n$.*
Output: *A permutation $\sigma \in G$ such that $(R_{\sigma_1}, v_{\sigma_1}, \ldots, R_{\sigma_n}, v_{\sigma_n})$ is lexicographically smallest.*

Such an algorithm can be achieved by making a combinatorial backtracking search over the prefix tree. At a node at level i we follow those edges leading to vertices whose markings σ_i make $(R_{\sigma_i}, v_{\sigma_i})$ lexicographically smallest. We keep a vector with the optimal permutation of R and v seen so far. Using this vector branches can be pruned if they cannot lead to an optimal permutation.

If stabilizers are small, which is often the case in our setting, and the group fits into memory, then the method described here works well. We refer to the field of computational group theory for other approaches, see [3] for references.

4 Oracles

Using different terminology the oracles for traversing normal fans of Minkowski sums of polytopes, secondary fans, Gröbner fans and tropical varieties are already present in the literature, see [6], [11],[5] and [2], respectively. The topic of this section is slight variations of these. Due to the size limit for this paper, we only discuss one of these oracles in detail, while briefly mentioning other possible variations.

We first consider the d-skeleton of the normal fan of a Minkowski sum of polytopes $P_1, \ldots, P_s$ whose vertices are given. This is a connected fan, and the link of a ridge with relative interior point ω is the d-skeleton of the normal fan of the Minkowski-sum of $\text{face}_\omega(P_1), \ldots, \text{face}_\omega(P_s)$. Modulo the lineality space the link is a collection of rays, which are the normals of the Minkowski sum of the faces. This is a general behavior; the computation becomes easier at the link – at least for $s = 2$, the Minkowski sum facets can be computed by a V-to-H conversion of the convex hull $\text{conv}((\text{face}_\omega(P_1) \times e_1) \cup \cdots \cup (\text{face}_\omega(P_s) \times e_s)) \subseteq \mathbb{R}^n \times \mathbb{R}^s$, which is also known as the *Cayley embedding* of $\text{face}_\omega(P_1), \ldots, \text{face}_\omega(P_s)$.

In the following we will explain how Gröbner fan computations can be restricted to cones of $\mathbb{R}^n$. It is important to note that a similar technique works for computing slices of secondary fans. One application of this can be found in the last paragraph of this paper.

4.1 The Gröbner fan

Recall that the maximal cones of $\Sigma(I)$ are in bijection with the marked reduced Gröbner bases $\{\mathcal{G}_\prec(I)\}_\prec$ where $\prec$ runs through all term orders. Inequalities for the Gröbner cone of $\mathcal{G}_\prec(I)$ can be read off from the exponent vectors of $\mathcal{G}_\prec(I)$. To make a change to another cone $O_C^{change}(R, v)$ through a ridge R with relative interior point ω and normal v, we compute the Gröbner basis with respect to the ordering given by $\omega + \varepsilon v$ with $\varepsilon > 0$ small (tie-broken in any way).

While the ε-perturbation is easy to handle in theory and practice with matrix term orders, the reader familiar with Gröbner bases will know that the above description is an oversimplification. One will not compute the $\omega + \varepsilon v$ Gröbner basis from scratch, but rather use the identity

$$\text{in}_{\omega+\varepsilon v}(I) = \text{in}_v(\text{in}_\omega(I))$$

to construct a Gröbner basis for I from one of $\text{in}_\omega(I)$. As for the Minkowski sum problem, the computation at the link becomes easier. See [5] and [8] for details.

We now explain how to restrict the Gröbner fan computation to a possibly lower-dimensional cone $D \subseteq \mathbb{R}^n$. One problem that we might face if the ideal is not homogeneous is that $\Sigma(I)$ is not complete and the usual restriction $\Sigma(I) \wedge \text{faces}(D)$ is not connected in codimension one – take for example $I = \langle x_1^2 x_2 + x_1 x_2^2 + 1 \rangle$ and $D = \{\omega \in \mathbb{R}^2 : \omega_1 + \omega_2 \leq 0\}$. There are several ways to get around this problem. Here, to keep the exposition simple, we will assume that I is homogeneous, and thus $\Sigma(I)$ complete.

Definition 1. *Let $I \subseteq k[x_1, \ldots, x_n]$ be an ideal and let $D \subseteq \mathbb{R}^n$ be a polyhedral cone. We define the* restriction $\Sigma(I)_D := \Sigma(I) \wedge \text{faces}(D)$.

The support of the restriction $\Sigma(I)_D$ is D.

Definition 2. *A ridge R in $\Sigma(I)_D$ is called* flippable *if* $\text{rel int}(R) \cap \text{rel int}(D)$ *is not empty.*

Lemma 1. *The restriction $\Sigma(I)_D$ is a pure fan connected in codimension* 1*. It is connected even if we only consider flippable ridges.*

Every maximal cone in $\Sigma(I)_D$ is of the form $\mathcal{C}_\prec(I) \cap D$ and we will represent such cone by $\mathcal{G}_\prec(I)$. This representation is not unique. We have described how the internal state of the oracle is stored, and will now explain how the oracle calls can be implemented:

Cone: The cone represented by $\mathcal{G}_\prec(I)$ can be computed as the intersection $\mathcal{C}_\prec(I) \cap D$.

Link: The link of a ridge has either one or two rays. One of these rays v is already known to us as the link of C at R in the oracle call and we need to decide if $-v$ is in the link. To check if R is flippable, it suffices to check if a relative interior point ω of R is in one of the facets of D.

Change: Let ω be a positive vector in the flippable ridge. A reduced Gröbner basis representing the neighbouring cone can be gotten by computing a Gröbner basis with respect to the term order given by $\omega + \varepsilon v$, tie-broken in any way. A more efficient way is to pass to the initial ideal $\mathrm{in}_\omega(I)$ first. See [5] and [8].

We note that for a facet $C \in \Sigma(I)_D$ it is easy to recover the initial ideal with respect to relative interior points of C. This was used in [4] for a method to check if a given generating set of I is a tropical basis.

There is still one problem that we need to address. Namely, how we get started, i.e. given I and D, how we compute a reduced Gröbner basis $\mathcal{G}_\prec(I)$ such that $\mathcal{C}_\prec(I) \cap D$ is maximal in $\Sigma(I)_D$. The solution is an application of matrix term orders. First we pick a vector $c_1 \in \text{rel int}((D + \mathcal{C}_0(I)) \cap \mathbb{R}^n_{\geq 0})$ and then extend c_1 to a basis $\{c_1, \ldots, c_d\}$ of $\mathrm{span}_\mathbb{R}(D)$. Then we extend this basis to a basis $\{c_1, \ldots, c_n\}$ of $\mathbb{R}^n$. The term ordering of the matrix with rows $c_1, \ldots, c_n$ gives the desired Gröbner basis. Alternatively, we consider $c = c_1 + \varepsilon c_2 + \ldots + \epsilon^{n-1} c_n$ and compute

$$\mathrm{in}_c(I) = \mathrm{in}_{c_n}(\mathrm{in}_{c_{n-1}}(\cdots \mathrm{in}_{c_1}(I) \cdots))$$

successively. This is the initial ideal for $\mathcal{G}_\prec(I)$. To construct $\mathcal{G}_\prec(I)$ we may repeatedly apply the Gröbner walk lifting procedure as it was done in [2, Algorithm 9].

Having specified the oracle, we may also apply the symmetric version of the traversal algorithm. The group G should be a symmetry group for $\Sigma(I)_D$.

It is tricky to extend Definition 1 and 2 to cover the non-homogeneous case and we shall only discuss one subtlety in this setting. Take $D = \mathbb{R}^n$. In this case, see [8], it is natural to allow only flips through facets with positive points in their interior, since this will guarantee usage of only allowable term orders at the ridge. Consider for example $I = \langle x^3 + y^3 + x^2y^2 \rangle$, which has a complete Gröbner fan with a ridge outside the strictly positive orthant. A priori, this ridge should not be considered flippable and the "flippable link" at that ridge should only consist of one ray, even though the geometric link consists of two. The only problem with this is that EnumerateRidge is called more than once for the same ridge in Algorithm 2. This does not change the correctness of the algorithm.

5 Comparison to reverse search

The memory-less reverse search method [1] can be used for traversing many types of full-dimensional fans – including Gröbner fans, see [8]. It works by traversing a spanning tree of the graph whose vertices are the facets and whose edges are the ridges of the fan. Symmetry can be exploited by restricting to a fundamental domain of the group action on $\mathbb{R}^n$, but still orbits whose cones touch the boundary of the fundamental domain may be computed more than ones. The reverse search has the drawback that in order to decide on the local structure of the traversal tree an O_C^{change} oracle call must be performed once for every ridge in a traversal. Therefore, for complicated oracles, it will often perform worse than Algorithm 2. On the other hand, the draw back of Algorithm 2 is that a vector in $\mathbb{Z}^{2n}$ needs to be stored for essentially every ridge-facet incidence pair and that the algorithm is not as easy to parallelize as reverse search, where interprocess communication is absent.

6 Implementation details

6.1 Handling geometric data

We briefly discuss how to compute properties of a cone given to us by an inequality description. The natural order of getting these properties is as follows: lineality space, span and dimension, facets, a relative interior point, rays.

The lineality space can be computed by Gauss elimination, while linear programming is needed for the span and the facets. Knowing the span of the cone is equivalent to knowing the implied equations of the inequalities. We refer to literature on the simplex algorithm. For the facets of a Gröbner cone, the inequality description is often highly redundant and a preprocessing step is useful. A relative interior point can be computed with linear programming and the computation of the rays can be reduced to an H-to-V conversion of a polytope.

Knowing the facets and rays it is a combinatorial task to extract the face lattice. Indeed given the set of rays of C contained in face A of C, we find all facets of A by, for each facet normal of C, picking the set of rays perpendicular to the normal. This may give lower dimensional faces of A, so we need to take the inclusion maximal sets of rays. They represent the facets of A.

Rather than computing the face lattice, it might be useful to know the orbits of all cones in a fan. We suggest keeping a list of canonical representatives for the orbits seen so far. Then we may run through the facets of F, and for each of these repeatedly apply the method of the previous paragraph, but for each newly computed face checking if its canonical representative has been seen already. A fast implementation of Algorithm 3 is useful at this point.

6.2 Software

The presented algorithm has been implemented according to a generic programming / object oriented paradigm in the software Gfan [10] and replaces old traversal strategies. Every oracle is derived from an abstract superclass. Starting from version 0.5 are features for computing restrictions of Gröbner fans and secondary fans. The oracle of Section 4.1 has been used in [4] to give a computer proof that the 4×4 minors of a 5×5 matrix are a tropical basis. In that paper also a tropical variety with a symmetry group of order 28800 was computed, explaining the need for handling symmetries. The Gfan software is written in C++, uses the libraries GMP [9] and cddlib [7] by default, and works like a Unix-style command line tool. In addition, Gfan can be linked to the floating point LP-solver SoPlex [12]. In this case LP-certificates will be lifted to $\mathbb{Q}$ using continued fractions, and checked. In case of failure, Gfan will fall back on cddlib.

Returning to Example 1, a two hour computation in Gfan gives the tropical variety $T(I_{\text{cusp}})$, exploiting a symmetry group of order 8. The 7-dimensional fan has a 3-dimensional lineality space and f-vector $(1, 1631, 7622, 11340, 5408)$. A total of 1431 ridges and 680 facets were processed. With a suitably prepared input file the computation can be done with the following command:

```
gfan_tropicaltraverse --symmetry < Icusp.startingcone
```

If the `--symmetry` option is left out the computation takes more than 15 hours. This order of speed-up is only expected for expensive oracles and symmetry groups that fit into memory. If the symmetry groups are sufficiently complicated and oracle calls are cheap, it is possible that all time saved on oracle calls is spent on computing canonical representatives.

We finish this paper by mentioning a few applications of Gfan and its algorithms to tropical geometry. In tropical geometry a natural question to ask is whether $\text{supp}(T(I_{\text{cusp}}))$ is the support of a subfan of the secondary fan $\mathcal{F}_1$ of the 2-dimensional Newton polytope of f. We may compute $\mathcal{F}_2 := \mathcal{F}_1 \wedge T(I_{\text{cusp}})$ by restricting the computation of $\mathcal{F}_1$ to each of the 680 facets. After this we pick a relative interior point from each facet of $\mathcal{F}_2$, and take the smallest subfan $\mathcal{F}_3 \subseteq \mathcal{F}_1$ containing these vectors. The question now is if $\text{supp}(\mathcal{F}_3) = \text{supp}(T(I_{\text{cusp}}))$. By construction, $\text{supp}(\mathcal{F}_3) \supseteq \text{supp}(T(I_{\text{cusp}}))$. The other inclusion can be checked by computing the restriction of $\Sigma(I_{\text{cusp}})$ to each cone of $\mathcal{F}_3$. Then we check for each corresponding initial ideal if it is monomial free. The polyhedral fan computations can be handled with Algorithm 2 and its implementation in Gfan. However, in our case none of this is needed since the vector (1) induces a regular subdivision whose secondary cone has dimension larger than the dimension of $\text{supp}(T(I_{\text{cusp}}))$.

References

1. David Avis and Komei Fukuda. A basis enumeration algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete Computational Geometry*, 8:295–313, 1992.
2. Tristram Bogart, Anders N. Jensen, David Speyer, Bernd Sturmfels, and Rekha R. Thomas. Computing tropical varieties. *J. Symbolic Comput.*, 42(1-2):54–73, 2007.
3. David Bremner, Mathieu Dutour Sikiric, and Achill Schürmann. Polyhedral representation conversion up to symmetries. *CRM proceedings*, 48:45–72, 2009.

4. Melody Chan, Anders Jensen, and Elena Rubei. The 4×4 minors of a $5 \times n$ matrix are a tropical basis. 2009, arXiv:0912.5264.
5. Stéphane Collart, Michael Kalkbrener, and Daniel Mall. Converting bases with the Gröbner walk. *J. Symbolic Comput.*, 24(3-4):465–469, 1997. Computational algebra and number theory (London, 1993).
6. Komei Fukuda. From the zonotope construction to the Minkowski addition of convex polytopes. *J. Symb. Comput.*, 38(4):1261–1272, 2004.
7. Komei Fukuda. *cddlib reference manual, cddlib Version 094b*. Swiss Federal Institute of Technology, Lausanne and Zürich, Switzerland, 2005. `http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html`.
8. Komei Fukuda, Anders Jensen, and Rekha Thomas. Computing Gröbner fans. *Mathematics of Computation*, 76:2189–2212, 2007.
9. Torbjörn Granlund et al. GNU multiple precision arithmetic library 4.3.1, 2009. `http://gmplib.org`.
10. Anders N. Jensen. Gfan, a software system for Gröbner fans and tropical varieties. Available at `http://www.math.tu-berlin.de/~jensen/software/gfan/gfan.html`.
11. Jörg Rambau. TOPCOM: Triangulations of point configurations and oriented matroids. *ZIB report*, 02-17, 2002.
12. Roland Wunderling. *Paralleler und objektorientierter Simplex-Algorithmus*. PhD thesis, Technische Universität Berlin, 1996. `http://www.zib.de/Publications/abstracts/TR-96-09/`.