**Wessel P.J. van Woerden**

# Perfect Quadratic forms: an Upper Bound and Challenges in Enumeration

**Master Thesis**

supervisor: Dr. Léo Ducas (CWI)

**Date exam: 30-08-2018**


**Mathematical Institute, Leiden University**
**Centrum Wiskunde & Informatica (CWI)**

## Abstract

**The lattice packing problem in dimension 9 is a long-standing open problem. We consider a classical algorithm by Voronoi to solve the lattice packing problem in any dimension, by enumerating all perfect quadratic forms. We show an improved upper bound on the number of non-similar perfect forms based on a volumetric argument and lattice reduction theory. Furthermore, we consider the challenges that arise when enumerating perfect forms, with a special look at dimension 9.**



## Acknowledgements

**This thesis was written with the support of my supervisor, Dr. Léo Ducas, whose valuable advice and guiding questions steered me in the right direction. I would like to thank Prof. Ronald Cramer for the opportunity to do an internship at CWI. Special thanks go to Prof. Achill Schürmann for his time and inspirational book that made me interested in the topic and to Dr. Mathieu Dutour Sikirić for his extensive software and ongoing collaboration on parallelising Voronoi's algorithm. I appreciate the help of Dr. Daniel Dadush, Koen de Boer, Maxime Plançon, Marieke Oudenes and Daan van Gent for general feedback or proofreading of my thesis.**


## Table of Contents

## 1. Introduction

1.1. **Introduction.** Mathematicians have long been inspired by the sphere packing problem. This problem asks how to pack $d$-dimensional unit balls in $\mathbb{R}^d$ such that their density, the proportion of $\mathbb{R}^d$ they fill, is maximized. In dimension 1 the sphere packing problem is trivial and in dimension 2 it was long expected that the hexagonal packing, the unique packing where each circle is touched by exactly 6 other circles, is optimal. The latter was first proved in 1910 by Thue [Thu10] and, after some doubts about the completeness of the proof, more rigorously around 1940 by Fejes Tóth [Fej42]. The sphere packing problem was much harder in dimension 3, for which Kepler [Kep10] had already conjectured an optimal packing in 1611. This packing consists of translated layers of hexagonal packings and is a well known way to stack oranges. Finally, in 1998 Hales [Hal05] famously proved Kepler's conjecture. Just as in dimension 2, there were some doubts about the rigour of the proof and in 2015 Hales, together with 21 coauthors, published a formal proof, that was verified by proof-checking software, to remove any doubt.

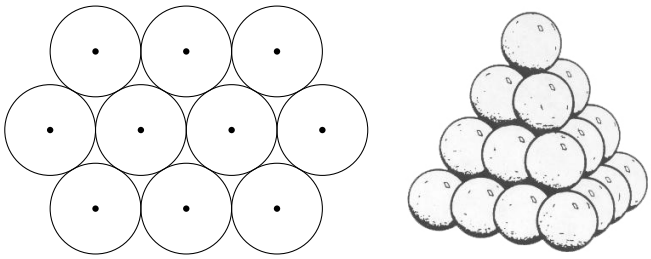

Figure 1. Optimal sphere packings in dimensions 2 and 3.

Quite paradoxically, the found optimal sphere packings are highly structured, whereas the general statement of the sphere packing problem is not. This lack of structure in the sphere packing problem makes it hard to solve. Consider the set $L \subset \mathbb{R}^d$ consisting of the centers of the unit balls in a sphere packing. For the aforementioned optimal packings in dimensions 2 and 3 this set $L$ is in fact a lattice, an additive discrete subgroup of $\mathbb{R}^d$. Therefore, a natural way to add more structure to the problem is to limit it to packings such that $L$ is a lattice.

This extra structure simplifies the problem significantly. Already in 1873 Korkine and Zolotarev [KZ73] solved the lattice packing problem up to dimension 5. In 1934, Blichfeldt [Bli35] proved the optimality of the root lattices $E_6, E_7$ and $E_8$ for dimensions 6, 7 and 8 using similar techniques as Korkine and Zolotarev. The famous Leech lattice $\Lambda_{24}$ is also an optimal lattice packing in dimension 24 [CK09]. In fact very recently it has been proved that $E_8$ [Via17] and $\Lambda_{24}$ [CKM+17] are even optimal for the general sphere packing problem. The densest known lattice in dimension 9 is the laminated lattice $\Lambda_9$, which was already conjectured by Korkine and Zolotarev to be optimal. However, almost 90 years after solving the lattice packing problem up to dimension 8, the conjecture in dimension 9 remains unproven.

In 1908, in his famous work [Voro8], Voronoi introduced an algorithm that solves the lattice packing problem in any dimension. Voronoi showed that any lattice with optimal packing density must be a so-called perfect lattice and his algorithm enumerates all perfect lattices up to similarity. However, this algorithm needs a lot of resources. Firstly, because the number of perfect lattices seems

to grow superexponentially in the dimension $d$. Secondly, because we need to solve the representation conversion problem for some highly degenerate polyhedral cones, which causes a combinatorial explosion. Therefore, it took until 2005 before Dutour Sikirić, Schürmann and Vallentin [DSSV07] were able to complete Voronoi's algorithm in dimension 8. They proved that there are exactly 10916 non-similar perfect 8-dimensional lattices. They also found more than 500.000 perfect 9-dimensional lattices, by running Voronoi's algorithm partially, with no end in sight.

1.2. **Overview.** In this thesis we examine the challenges that arise when trying to complete Voronoi's algorithm. In particular we take a practical look at Voronoi's algorithm in dimension 9. After some basic definitions in Section 2 we will introduce in Section 3 an upper bound of $e^{O(d^2 \log d)}$ on the number of non-similar perfect $d$-dimensional lattices. This improves on the recently shown asymptotic upper bound of $e^{O(d^3 \log d)}$ by Bacher [Bac17].

We introduce Voronoi's algorithm in Section 4 and we give an overview of the sub problems that need to be solved. In Sections 5 and 6 we consider some implementation improvements for two steps of Voronoi's algorithm. These improvements are necessary due to the large number of perfect 9-dimensional lattices. Using these improvements we were able to find much more 9-dimensional perfect lattices than found so far. In fact we consistently find around 1.5 million new perfect 9-dimensional lattices each day on a single core, which is three times the number of perfect lattices known so far.

In Section 7 we discuss how to deal with symmetries. In particular we discuss recent improvements on finding a canonical representative of an orbit under a group action. We show in Section 8 that these improvements are useful to solve the representation conversion problem under symmetry, which is needed to deal with some very degenerate cones arising in Voronoi's algorithm. We introduce in Section 9 a new and efficient algorithm to enumerate all faces of a polyhedral cone up to symmetry. In Section 10 we take a look at the two hardest instances of the representation conversion problem that arise, as far as we know, when running Voronoi's algorithm in dimension 9. We solve one of these instances without much computation.

The results and the analysis of partially running Voronoi's algorithm in dimension 9 are shown in Section 11. Using knowledge about the found perfect forms and an earlier solved instance of the representation conversion problem in Section 10, we try to estimate the number of 9-dimensional perfect forms with heuristic arguments. This estimate indicates that parallelisation of Voronoi's algorithm is necessary, thus we introduce an easy and well performing partitioning function for perfect forms.

1.3. **Contributions.** Here we show a list of the major contributions of this thesis.

- An upper bound of $e^{O(d^2 \log(d))}$ on the number of non-similar $d$-dimensional perfect forms. Using completely different techniques, this improves on the recent upper bound of $e^{O(d^3 \log(d))}$ by Bacher [Bac17]. See Section 3.
- Optimizations and a thorough complexity overview on the problem of determining contiguous perfect forms. See Section 5.

- An overview and correctness check of a recent algorithm by Dutour Sikirić to obtain a canonical perfect form [DS18]. In collaboration with Dutour Sikirić. See Section 6.3.
- Optimizations to the dual description problem under symmetry by using the state of the art algorithms in dual description and canonical orbit representatives. See Section 8.5.
- An adaptation of the geometric face enumeration algorithm by Fukuda, Liebling and Margot [FLM97] to exploit available symmetry. See Section 9.
- A look at the two most degenerate cones $\mathcal{V}(Q_{129})$ and $\mathcal{V}(Q_{\Lambda_9})$ that appear in Voronoi's algorithm in dimension 9. In particular we derive an orbit description of the facets of the highly degenerate cone $\mathcal{V}(Q_{129})$ from the already known orbit description of the facets of the cone $\mathcal{V}(Q_{E_8})$. See Section 10.
- Using the optimizations above, we found $\geq$ 23.000.000 perfect 9-dimensional forms, while so far only 500.000 were known. Based on the results we heuristically derive an estimation on the number of 9-dimensional non-similar perfect forms. We introduce an efficient partitioning function for perfect forms, necessary for parallelisation, that performs very well on the known set of 9-dimensional perfect forms. See Section 11.
- Beyond the scope of this thesis, there is a collaboration with M. Dutour Sikirić to implement a parallelised version of Voronoi's algorithm using the standardized Message Passing Interface (MPI). Using the found optimizations in this thesis and by running it on a super computer, we have hope to finally finish Voronoi's algorithm in dimension 9 and thereby solve the 9-dimensional lattice packing problem.

## 2. Preliminaries

In this section we recall some classical definitions and results on quadratic forms, lattices, lattice reduction theory, perfect forms and polyhedral cones. The definitions and notation will be used in the remainder of this thesis. Most of the notation originates from the extensive work on positive definite quadratic forms by Schürmann [Sch09].

2.1. **Basic notations.** We denote the sets of integers, rationals and reals by $\mathbb{Z}, \mathbb{Q}$ and $\mathbb{R}$ respectively. With $\mathbb{R}_{\geq 0}$ and $\mathbb{R}_{>0}$ we denote the set of all non-negative respectively positive reals. The set of integers $\{1, 2, \ldots, m\}$ is denoted by $[m]$ for any integer $m \geq 1$. $\mathcal{P}([m])$ is the power set of $[m]$, consisting of all subsets of $[m]$. Unless otherwise stated a vector $v \in \mathbb{R}^d$ is a column vector. The transpose of a vector $v \in \mathbb{R}^d$ or of a matrix $A \in \mathbb{R}^{d\times d}$ is denoted by $v^t$ or $A^t$ respectively. All products between vectors or matrices are considered matrix products. In particular $v^t w = \sum_{i=1}^{d} v_i \cdot w_i$ and $vw^t = (v_i \cdot w_j)_{i,j} \in \mathbb{R}^{d\times d}$ for column vectors $v, w \in \mathbb{R}^d$. The trace and determinant of a square matrix $A$ are denoted by $\mathrm{Tr}(A)$ and $\det(A)$ respectively. By $\mathrm{GL}_d(R)$ we denote the multiplicative matrix group consisting of all invertible $d \times d$ matrices over the ring $R$.

For a permutation group $G \subset \mathrm{Sym}_m$ the action on $[m]$ by a permutation $\sigma \in G$ is given by $x \mapsto \sigma \cdot x := \sigma(x)$. The action of $G$ on $\mathcal{P}([m])$ is given by $X \mapsto \sigma \cdot X := \{\sigma(x) : x \in X\}$. We say that $G$ acts invariantly on a set $S \subset \mathcal{P}([m])$ if $\sigma \cdot X \in S$ for all $X \in S$ and $\sigma \in G$. The orbit of an element $x$ under the action of $G$ is denoted by $\mathrm{Orb}(G, x)$. The stabilizer of an element $x$ under the action of $G$ is denoted by $\mathrm{Stab}(G, x) \subset G$.

For a polytope or cone $P$ we denote the interior by $\mathrm{Int}(P)$. If $P \subset \mathbb{R}^n$ is bounded we denote the standard $n$-dimensional volume of $P$ by $\mathrm{Vol}(P)$.

2.2. **Quadratic forms.** A (real) quadratic form in $d \geq 1$ variables is a function

$$\begin{aligned} Q : \mathbb{R}^d &\to \mathbb{R}, \\ x &\mapsto Q[x] := x^t Q x, \end{aligned}$$

for a symmetric real matrix $Q \in \mathbb{R}^{d\times d}$. The space of all real quadratic forms is denoted by

$$\mathcal{S}^d = \{Q \in \mathbb{R}^{d\times d} : Q^t = Q\}.$$

Note that $\mathcal{S}^d$ is an $n := \binom{d+1}{2}$-dimensional real vector space, which is a Euclidean space when endowed with the standard trace inner product

$$\langle P, Q\rangle := \mathrm{Tr}(P^t Q) = \sum_{i,j\in[d]} P_{ij} Q_{ij}.$$

By cyclicity of the trace, we have $x^t Q x = \langle Q, x x^t\rangle$. A natural isometry from $\mathcal{S}^d$ to the canonical Euclidean space $\mathbb{R}^n$ is given by:

$$\phi : \mathcal{S}^d \to \mathbb{R}^n : Q \mapsto (q_{ij})_{i\leq j}$$

where $q_{ii} := Q_{ii}$ and $q_{ij} := \sqrt{2} Q_{ij} = \sqrt{2} Q_{ji}$ for $i < j$. Indeed we have

$$\langle \phi(P), \phi(Q)\rangle = \langle P, Q\rangle.$$

Most geometric objects in the remainder of this thesis are defined in the canonical Euclidean space $\mathbb{R}^n$. When speaking about these objects in $\mathcal{S}^d$, or doing computations with them, we implicitly use the isometry mentioned above.

Moreover, we consider the cone of positive definite quadratic forms (PQFs)

$$\mathcal{S}^d_{>0} = \{Q \in \mathcal{S}^d : Q \text{ is positive definite}\},$$

its closure

$$\mathcal{S}^d_{\geq 0} = \{Q \in \mathcal{S}^d : Q \text{ is positive semidefinite}\},$$

and finally its historically named [Namo6] rational closure

$$\tilde{\mathcal{S}}^d_{\geq 0} = \operatorname{cone}\{xx^t : x \in \mathbb{Z}^n\} = \left\{ \sum_{x \in \mathbb{Z}^n} c_x \cdot xx^t \in \mathcal{S}^d : c_x \in \mathbb{R}_{\geq 0} \right\}.$$

In particular we have $\mathcal{S}^d_{>0} \subset \tilde{\mathcal{S}}^d_{\geq 0} \subset \mathcal{S}^d_{\geq 0}$. See [Scho9] for another characterization of $\tilde{\mathcal{S}}^d_{\geq 0}$ which makes the first inclusion clear.

2.3. **Arithmetical equivalence.** Two quadratic forms are arithmetically equivalent if they lie in the same orbit under the action $Q \mapsto U^t Q U$ of the group

$$\mathrm{GL}_d(\mathbb{Z}) = \{U \in \mathbb{Z}^{d \times d} : |\det U| = 1\}.$$

If $Q$ and $Q'$ are positive definite quadratic forms, then we call $Q$ and $Q'$ similar if and only if $Q$ is arithmetically equivalent to $\alpha Q'$ for some $\alpha > 0$.

2.4. **Positive definite quadratic forms.** For any PQF $Q \in \mathcal{S}^d_{>0}$ there exists a smallest real number $r > 0$ for which $Q[x] = r$ has an integral solution. We define this number as the arithmetical minimum denoted by

$$\lambda(Q) := \min_{x \in \mathbb{Z}^d \setminus \{0\}} Q[x].$$

More generally, we define for $i \in [d]$ the $i$-th succesive minima $\lambda_i(Q)$ as

$$\begin{aligned}\lambda_i(Q) := \inf\{\lambda > 0 : \exists\ \mathbb{R}\text{-linearly independent } x_1, \ldots, x_i \in \mathbb{Z}^n \setminus \{0\} \\ : Q[x_j] \leq \lambda \ \text{ for all } j \in [i]\},\end{aligned}$$

where the infinum is in fact a minimum. Note that $\lambda_1(Q) = \lambda(Q)$ and $\lambda_i(\alpha Q) = \alpha \lambda_i(Q)$ for $\alpha \in \mathbb{R}_{>0}$ . These quantities are invariant under arithmetical equivalence, because $Q[Ux] = (U^t Q U)[x]$. So under the assumption that $\lambda(Q) = \lambda(Q')$ for PQFs $Q, Q' \in \mathcal{S}^d_{>0}$, the notions of similarity and arithmetical equivalence coincide.

For a PQF $Q \in \mathcal{S}^d_{>0}$ let $x_1, \ldots, x_d \in \mathbb{Z}^d$ be linear independent such that $Q[x_i] = \lambda_i(Q)$ for all $i \in [d]$. Then by Hadamard's inequality

$$\det(Q) \leq \prod_{i=1}^{d} Q[x_i] = \prod_{i=1}^{d} \lambda_i(Q).$$

Every PQF $Q \in \mathcal{S}^d_{>0}$ can be written as a sum of squares of linear terms. Its so-called Lagrange expansion is for all $y = (y_1, \ldots, y_d) \in \mathbb{R}^d$ given by

$$Q[y] = \sum_{i=1}^{d} A_j \left( y_i - \sum_{j=i+1}^{d} \alpha_{ij} y_j \right)^2$$

with unique $A_i \in \mathbb{R}_{>0}$ and $\alpha_{ij} \in \mathbb{R}$ for all $i \in [d]$ and $j \in \{i+1, \ldots, d\}$.

We define the set of minimal vectors of a PQF $Q \in \mathcal{S}^d_{>0}$ as

$$\operatorname{Min} Q := \{x \in \mathbb{Z}^d : Q[x] = \lambda(Q)\}.$$

Note that if $Q' = U^t Q U$, then $\operatorname{Min} Q = U \operatorname{Min} Q'$. We also define what is called the Voronoi domain $\mathcal{V}(Q)$ of a PQF $Q \in \mathcal{S}_{>0}$ as

$$\mathcal{V}(Q) := \operatorname{cone}(\{xx^t : x \in \operatorname{Min} Q\}) = \left\{ \sum_{x \in \operatorname{Min} Q} c_x x x^t \in \mathcal{S}^d : c_x \in \mathbb{R}_{\geq 0} \right\} \subset \mathcal{S}^d_{\geq 0}.$$

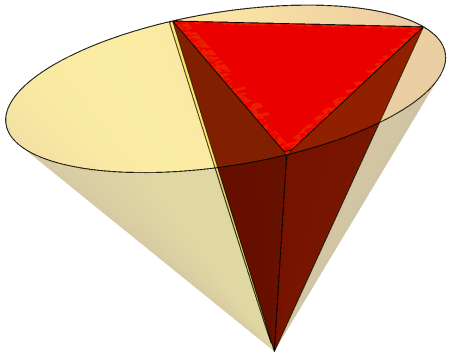

FIGURE 2. Voronoi domain of the PQF $\left(\begin{smallmatrix} 2 & 1 \\ 1 & 2 \end{smallmatrix}\right)$ in the cone $\mathcal{S}^2_{\geq 0}$.

For a PQF $Q \in \mathcal{S}^d_{>0}$ we define the dual PQF as the inverse matrix $Q^{-1} \in \mathcal{S}^d_{>0}$; this coincides with lattice duality. Note that if the PQFs $Q, Q' \in \mathcal{S}^d_{>0}$ are arithmetically equivalent, then $Q^{-1}$ and $(Q')^{-1}$ are also arithmetically equivalent (by $U^{-t}$). There are several metric relations between the PQFs $Q$ and $Q^{-1}$, known as transference theorems. In particular for the succesive minima we have the bound

$$\lambda_i(Q)\lambda_{d-i+1}(Q^{-1}) \leq d^2$$

for all $i \in [d]$ by Banaszczyk [Ban93].

2.5. **Lattices and HKZ reduction.** A rank $k$ lattice in $d$-dimensional space is the discrete additive group $L = L(B) = B\mathbb{Z}^k \subset \mathbb{R}^d$, where the matrix $B$ consists of linear independent column vectors $b_1, \ldots, b_k \in \mathbb{R}^d$. We call $B$ a basis of the lattice $L$. The quadratic form associated to a full rank lattice with basis $B \in \mathrm{GL}_d(\mathbb{R})$ is given by its Gram matrix $Q = B^t B = (\langle b_i, b_j \rangle)_{i,j} \in \mathcal{S}^d_{>0}$. However, the PQF $Q = B^t B$ is not only associated to the basis $B$, but precisely to all of its orthonormal transformations $OB$ with $O \in \mathcal{O}_d(\mathbb{R}) = \{O \in \mathbb{R}^{d \times d} : O^t O = I_d\}$. So there is a bijection between the set $\mathcal{S}^d_{>0}$ of PQFs and the set $\mathcal{O}_d(\mathbb{R}) \backslash \mathrm{GL}_d(\mathbb{R})$ of lattice bases up to orthonormal transformations. In particular a PQF $Q$ is associated to the lattice basis $C$ where $C$ is the upper triangular matrix from the Cholesky decomposition $Q = C^t C$.

For a lattice with basis $B$, all bases are of the form $B' = BU$ with $U \in \mathrm{GL}_d(\mathbb{Z})$. Note that for the associated PQFs $Q$ and $Q'$ respectively we have

$$Q' = (BU)^t (BU) = U^t (B^t B) U = U^t Q U$$

and thus $Q$ and $Q'$ are arithmetically equivalent. So the set $\mathcal{S}^d_{>0} / \mathrm{GL}_d(\mathbb{Z})$ of PQFs up to arithmetical equivalence is in one-to-one correspondence with the set $\mathcal{O}_d(\mathbb{R}) \backslash \mathrm{GL}_d(\mathbb{R}) / \mathrm{GL}_d(\mathbb{Z})$ of lattices up to isometries.

For a basis $B = [b_1, \ldots, b_k]$ we define the Gram-Schmidt orthogonalization $B^\dagger = [b_1^\dagger, \ldots, b_k^\dagger]$ of $B$ recursively with $b_1^\dagger = b_1$ and for $2 \leq i \leq k$ as

$$b_i^\dagger = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^\dagger,$$

where for all $1 \leq j < i \leq k$

$$\mu_{i,j} = \frac{\langle b_i, b_j^\dagger \rangle}{\langle b_j^\dagger, b_j^\dagger \rangle}.$$

The study of finding a basis of a lattice $L$ with some desired properties is called reduction theory. Basis reduction algorithms attempt to find a basis where the basis vectors are short and somewhat orthogonal. Reduction algorithms can differ in complexity and in the guaranteed properties a reduced basis has. An important part of the definition of a reduction is that for every lattice there exists such a reduced basis. A famous polynomial time reduction algorithm is the LLL-algorithm by A. Lenstra, H. Lenstra and Lovász [LLL82]. The LLL-algorithm is fast, but only guarantees to find a shortest vector of a full rank lattice $L \subset \mathbb{R}^d$ up to a factor of $2^{O(d)}$. For us the LLL-algorithm is mostly useful in practice.

For a more theoretical application we consider Hermite-Korkine-Zolotarev (HKZ) reduction [KZ72] of a rank $k$ lattice with basis $B = [b_1, \ldots, b_k] \in \mathbb{R}^{k \times d}$. For $i \in [k]$ we denote by $\pi_i : \mathbb{R}^n \to \operatorname{span}(b_1, \ldots, b_{i-1})^\perp$ the projection orthogonal to $b_1, \ldots, b_{i-1}$. Note that $\pi_i(b_i) = b_i^\dagger$. The basis $B$ of $L$ is called HKZ-reduced if:

(1) $b_i^\dagger$ is a shortest non-zero vector of the rank $k - i + 1$ lattice $\pi_i(L)$;

(2) $|\mu_{i,j}| \leq \frac{1}{2}$ for all $1 \leq j < i \leq k$.

We need the following property of a HKZ-reduced basis.

**Lemma 1** ([LLS90]). *Suppose that $B = [b_1, \ldots, b_k] \in \mathbb{R}^{k \times d}$ is a HKZ-reduced basis of the lattice $L$, and let $Q = B^t B$, then*

$$Q_{ii} = \|b_i\|^2 \leq \frac{i+3}{4} \lambda_i(Q) \qquad \text{for } 1 \leq i \leq k.$$

*Proof.* Let $i \in [k]$. There are $i$ linearly independent vectors of squared length at most $\lambda_i(Q)$ in $L$. So under the projection $\pi_i$, at least one of them maps to a non-zero vector. As an orthogonal projection only shortens vectors and $b_i^\dagger$ is a shortest vector of $\pi_i(L)$, we get $\left\|b_i^\dagger\right\|^2 \leq \lambda_i(Q)$. Now we can conclude by $|\mu_{i,j}| \leq \frac{1}{2}$, that

$$\begin{aligned} \|b_i\|^2 &= \left\| b_i^\dagger + \sum_{j=1}^{i-1} \mu_{i,j} b_j^\dagger \right\|^2 = \left\|b_i^\dagger\right\|^2 + \sum_{j=1}^{i-1} |\mu_{i,j}|^2 \left\|b_j^\dagger\right\|^2 \\ &\leq \lambda_i(Q) + \frac{1}{4} \sum_{j=1}^{i-1} \lambda_j(Q) \leq \frac{i+3}{4} \lambda_i(Q). \end{aligned}$$

□

2.6. **Lattice packing and the Hermite invariant.** An important property of a PQF $Q \in \mathcal{S}_{>0}^d$ is the Hermite invariant

$$\gamma(Q) = \frac{\lambda(Q)}{(\det Q)^{1/d}},$$

which is invariant under scaling and arithmetical equivalence. Let $L$ be a lattice and let $Q$ be a PQF corresponding to some basis of $L$. The lattice packing of such a lattice $L$ consists of balls with radius $\frac{1}{2}\sqrt{\lambda(Q)}$. Furthermore any fundamental area of $\mathbb{R}^d$ modulo $L$ has volume $\sqrt{\det(Q)}$. Therefore, as illustrated in Figure 3, the packing density $\delta(L)$ of the lattice $L$ is given by

$$\delta(L) = \gamma(Q)^{d/2} \cdot \frac{\mathrm{vol}(B_1^d)}{2^d}.$$

with $B_1^d$ the $d$-dimensional unit ball.

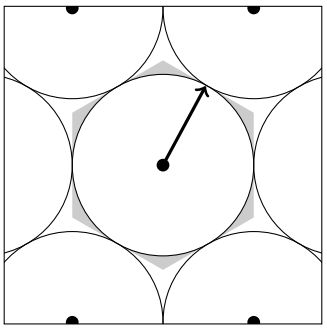

FIGURE 3. The packing density of a lattice $L$ equals the volume of a single ball divided by the volume of a fundamental area modulo $L$.

The lattice packing problem in dimension $d$ asks to find the supremum of the packing density over all $d$-dimensional full rank lattices. Solving the lattice packing problem in dimension $d$ is thus equivalent to determining the Hermite constant

$$\mathcal{H}_d = \sup_{Q \in \mathcal{S}^d_{>0}} \gamma(Q) = \sup_{Q \in \mathcal{S}^d_{>0}} \frac{\lambda(Q)}{(\det Q)^{1/d}}.$$

There exist several upper bounds for the Hermite constant. Although a linear upper bound exists by Blichfeldt [Bli29], we only mention the upper bound $\mathcal{H}_d \leq (4/3)^{(d-1)/2}$. This upper bound follows, for example, by repeated use of Mordell's inequality, that states

$$\mathcal{H}_d \leq \mathcal{H}_{d-1}^{(d-1)/(d-2)}$$

for $d \geq 3$. The bound follows from $\mathcal{H}_2 = \sqrt{4/3}$.

2.7. **Perfect forms.** The form $Q$ is called perfect if the set of equations

$$\langle Q', xx^t \rangle = x^t Q' x = \lambda(Q) \text{ for every } x \in \mathrm{Min} Q$$

has the unique solution $Q' = Q$ among $Q' \in \mathcal{S}^d$. So a perfect form is uniquely determined by its minimal vectors. In particular a PQF $Q \in \mathcal{S}^d_{>0}$ being perfect is equivalent to $\mathcal{V}(Q)$ having full dimension $n$ in $\mathcal{S}^d$. This also implies for a perfect form $Q$ that $\mathrm{Min}\, Q$ spans $\mathbb{R}^d$ over $\mathbb{R}$ and thus in particular $\lambda(Q) = \lambda_1(Q) = \ldots = \lambda_d(Q)$. So for a perfect form $Q$ we have

$$\det(Q) \leq \prod_{i=1}^{d} \lambda_i(Q) = \lambda(Q)^d.$$

Another consequence is that any perfect form $Q$ can be scaled by $\alpha \in \mathbb{R}_{>0}$ such that $\alpha Q$ is rational. Namely because the coefficients of a perfect form $Q$ are determined by rational linear equations, given by its minimal vectors, if $\lambda(Q)$ is rational. For a perfect form $Q$ with $\lambda(Q) = 1$, we call the minimal $s \in \mathbb{Z}_{>0}$ such that $sQ$ is integral, the scale of $Q$.

**Theorem 2.** *(Voronoi [Voro8]) The number of non-similar perfect forms in any fixed dimension $d$ is finite.*

Improving on Theorem 2 we prove in Section 3 an absolute upper bound to the number of non-similar perfect forms in any fixed dimension $d$.

A PQF $Q$ is called extreme if the Hermite invariant $\gamma$ attains a local maximum at $Q$ in $\mathcal{S}^d_{>0}$. A PQF $Q$ is called eutactic if there exist $\lambda_x > 0$ for all $x \in \operatorname{Min} Q/\{\pm 1\}$ such that

$$Q^{-1} = \sum_{x \in \operatorname{Min} Q/\{\pm 1\}} \lambda_x x x^t.$$

If we weaken the constraint to $\lambda_x \geq 0$ a PQF is called semi-eutactic. To check algorithmically if a PQF $Q$ is eutactic we solve the following linear program with $B \in \mathbb{R}^{n \times (|\operatorname{Min} Q|/2)}$ having columns $\phi(xx^t)$ for $x \in \operatorname{Min} Q/\{\pm 1\}$:

| | | |
|---|---|---|
| **Maximize:** | $y$ | |
| **Subject to:** | $B\lambda = \phi(Q^{-1})$, | |
| | $\lambda_x - y \geq 0$ | for all $x \in \operatorname{Min} Q/\{\pm 1\}$. |

Then $Q$ is eutactic if and only if this program has a strictly positive optimal value $y > 0$. If $\lambda = 0$, then $Q$ is semi-eutactic. In Section 4.1 we see that a PQF is extreme if and only if it is perfect and eutactic, so the above linear program applied to a perfect form determines if it is extreme.

2.8. **Polyhedra and the face lattice.** We consider convex polyhedra $\mathcal{C} \subset \mathbb{R}^{n-1}$. For a thorough treatment and proofs, we refer to Ziegler [Zie12]. By homogenization we can always assume that $\mathcal{C}$ is in fact a pointed polyhedral cone $\mathcal{C} \subset \mathbb{R}^n$, which we will call cone from now on. By definition a cone can be represented by an intersection of half-spaces

$$\mathcal{C} = P(A, 0) := \{x \in \mathbb{R}^n : Ax \geq 0\}$$

for some $A \in \mathbb{R}^{n \times m}$. Here $Ax \geq b$ is notation for $m$ inequalities $a_i \cdot x \geq b_i$ where $a_1, \ldots, a_m$ are the rows of $A$. This representation by half-spaces is also called the $\mathcal{H}$-representation.

By the representation theorem of cones, an equivalent way to describe $\mathcal{C}$ is as

$$\mathcal{C} = \operatorname{cone}(Y) := \left\{ \sum_{i=1}^{l} c_i y_i : c_i \in \mathbb{R}_{\geq 0} \right\}$$

for some $Y = [y_1, \ldots, y_l] \in \mathbb{R}^{n \times l}$. Sets of the form $\{c_i y_i : c_i \in \mathbb{R}_{\geq 0}\} \subset \mathcal{C}$ are called rays of $\mathcal{C}$. This representation by vertices and rays, in our case only rays, is also called the $\mathcal{V}$-representation.

An inequality $c^t x \geq 0$ with column vectors $c \in \mathbb{R}^n$ is called valid for $\mathcal{C}$ if it is satisfied for all $x \in \mathcal{C}$. A face of $\mathcal{C}$ is any set of the form

$$F = \mathcal{C} \cap \{x \in \mathbb{R}^d : c^t x = 0\},$$

where $c^t x \geq 0$ is valid for $\mathcal{C}$. The dimension $\dim(F)$ of a face $F$ is the dimension of its affine hull $\text{aff}(F)$. In our case of cones this is the dimension of the smallest linear subspace containing $F$. We call a face of dimension $k$ a $k$-face. Note that by this definition 0 and $\mathcal{C}$ are also faces of $\mathcal{C}$. The faces of dimensions $0, 1, \dim(\mathcal{C}) - 2$, and $\dim(\mathcal{C}) - 1$ are called vertices, extreme rays, ridges and facets respectively. Note that $0 \in \mathbb{R}^n$ is the unique vertex of a cone.

Observe that if $A$ consists of the minimal number of inequalities to describe $\mathcal{C}$, then the facets of $\mathcal{C} = P(A, 0)$ correspond exactly to those inequalities. In the same way if $Y$ consists of the minimal number of generators such that $\mathcal{C} = \text{cone}(Y)$, then these generators correspond exactly to the extreme rays of $\mathcal{C}$.

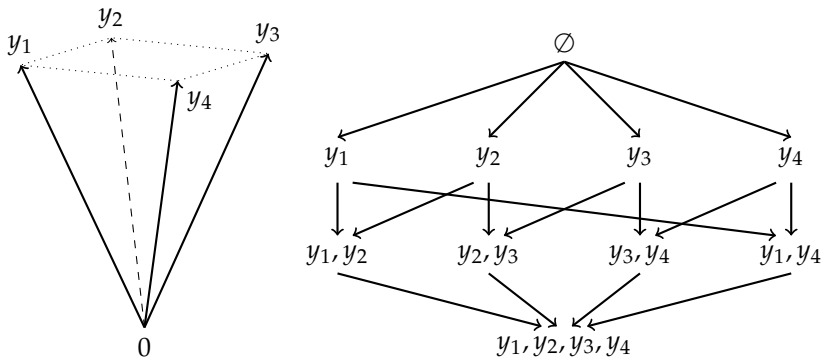


FIGURE 4. Face lattice of a cone generated by 4 extreme rays $y_1, y_2, y_3, y_4$. The faces are indicated by the extreme rays generating them.

The face lattice of $\mathcal{C}$ is the set of all faces of $\mathcal{C}$ together with the partial ordering obtained from inclusion. The name 'lattice' should not be confused with the terminology of lattices as discrete additive groups. A lattice is a poset (partially ordered set) with some special properties. We show these properties in terms of the face lattice. The face lattice is most clearly represented by a Hasse diagram. This is a directed acyclic graph where each vertex corresponds to a face. There is a directed edge from vertices $F$ to $F'$ if $F \subset F'$ and the only faces $F \subset G \subset F'$ are $G = F$ and $G = F'$. Then $F \subset F'$ if and only if there exists a path from $F$ to $F'$ in the Hasse diagram. For clarity all $k$-faces are often represented on a single level, where the levels go from 0 to $\dim(\mathcal{C})$.

We mention some important properties of faces $F, F'$ of a cone $\mathcal{C}$:

(1) $F$ is itself a cone.

(2) $F \cap F'$ is also a face of $\mathcal{C}$.

(3) There exists a unique minimal face $G$ of $\mathcal{C}$ such that $F, F' \subset G$.

(4) The faces of $F$ are exactly the faces of $\mathcal{C}$ that are contained in $F$.

(5) If $\dim(F) = k < \dim(\mathcal{C})$, then $F$ is the intersection of the facets of $\mathcal{C}$ that contain $F$.

(6) If $\dim(F) = k > 0$, then $F$ is the cone generated by the extreme rays of $\mathcal{C}$ that are contained in $F$.

Note the symmetry between some of these properties. This is an immediate consequence of conic duality. For every cone $\mathcal{C} \subset \mathbb{R}^n$ there exists a dual, not necessarily pointed, cone $\mathcal{C}^* = \{y \in \mathbb{R}^n : \langle x, y \rangle \geq 0 \text{ for all } x \in \mathcal{C}\}$, for which the

roles of facets and extreme rays are swapped. If $\mathcal{C}$ is a full dimensional cone, then $\mathcal{C}^*$ is also a full dimensional cone.

For cones $\mathcal{C}$ this particularly means that the $\mathcal{V}$-representation $\mathcal{C} = P(A, 0)$ gives the $\mathcal{H}$-representation $\mathcal{C}^* = \text{cone}(A^t)$. Furthermore $(\mathcal{C}^*)^* = \mathcal{C}$ and so we can similarly convert an $\mathcal{H}$-representation of $\mathcal{C}$ to a $\mathcal{V}$-representation of $\mathcal{C}^*$. See Figure 5 for an overview. We also get a one-to-one inclusion reversing correspondence between $k$ dimensional faces of $\mathcal{C}$ and $n-k$ dimensional faces of $\mathcal{C}^*$. In particular the face lattice stays the same under this correspondence, except for the reversal of the inclusions. As a result for our algorithms involving faces, we can assume without loss of generality, by switching to the dual, that the cone is given either by its $\mathcal{V}$ or $\mathcal{H}$-representation. Most algorithms are more easily explained in one of the representations, so we will switch freely between the two.

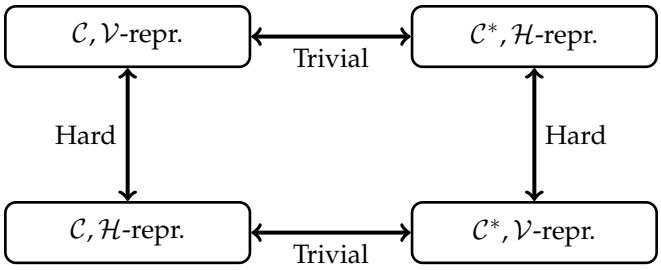


FIGURE 5. Overview of conversion between representations.

2.9. **Face-defining sets and symmetries of cones.** By properties (5) and (6) of a face mentioned in Section 2.8 we can define so-called face-defining sets. The definition of face-defining sets depends on the usage of either an $\mathcal{H}$ or a $\mathcal{V}$-representation. However, by duality, we do not need to do anything different between these two representations. Let $A \in \mathbb{R}^{m \times n}$ have rows $a_1, \ldots, a_m \in \mathbb{R}^n$ and suppose that $\mathcal{C}$ has either a non-redundant $\mathcal{H}$-representation $Ax \geq 0$ or a $\mathcal{V}$-representation $\text{cone}(A^t)$ respectively. We associate the facets and extreme rays by their index $i \in [m]$ respectively. Let $H \subset \mathcal{P}$ be a face of $\mathcal{C}$. Then the face-defining set $F_H \subset [m]$ of $H$ is the set of facets that fully contain $H$ or the set of extreme rays that are fully contained in $H$ respectively. Given a face-defining set $F_H$, the unique face corresponding to it in an $\mathcal{H}$-representation is

$$H = \{x \in \mathcal{C} : a_i \cdot x = 0 \text{ for all } i \in F_H\},$$

and in a $\mathcal{V}$-representation

$$H = \text{cone}(\{a_i^t : i \in F_H\}).$$

So we have a one-to-one correspondence between faces and face-defining sets of a cone $\mathcal{C}$. Now how do we check if $F \subset [m]$ is a face-defining set? This is more clear in the setting of an $\mathcal{H}$-representation as every intersection of facets $F$ defines a face $H \subset \mathcal{C}$. What we then need to check is if $F$ is exactly the face-defining set $F_H$ corresponding to $H$. Note that at least $F \subset F_H$ and that a facet is present in $F_H$ if and only if $H$ is fully contained in this facet. If $H$ is not fully contained in a facet $i \in [m] \setminus F$, then there must exist an $x \in H \subset \mathcal{C}$ such that $a_i \cdot x > 0$. As $H$ is closed under addition, we can do the above check for all facets $i \in [m] \setminus F$ at the

same time. So $F \subset [m]$ is face-defining for $\mathcal{C}$ if and only if there exists an $x \in \mathbb{R}^n$ such that

$$\begin{aligned} a_i \cdot x &= 0 && \text{for all } i \in F \\ a_i \cdot x &> 0 && \text{for all } i \in [m] \setminus F. \end{aligned}$$

This can algorithmically be checked by a linear program solver. See Section 9.1 for more information on this. The above check also works for the $\mathcal{V}$-representation although this is a bit harder to see. The existence of such an $x$ shows that $a_i^t \notin H := \text{cone}(\{a_j^t : j \in F\})$ for all $i \in [m] \setminus F$ as $x$ is orthogonal to $H$. Furthermore this $x$ gives us a valid inequality that shows that $H$ is indeed a face of $\mathcal{C}$. In particular a set $F \subset [m]$ is face-defining for $\mathcal{C} = P(A, 0)$ if and only if it is face-defining for $\mathcal{C}^* = \text{cone}(A^t)$.

In the remainder of this thesis we will represent faces by their face-defining set, which are defined by an $\mathcal{H}$ or a $\mathcal{V}$-representation depending on the context. Note however that the correspondence with the $\mathcal{H}$-representation is inclusion reversing. In particular the face lattice of $\mathcal{C}$ can be represented by face-defining sets. Let $\mathcal{F}_k$ be the set of $k$-dimensional faces of $\mathcal{C}$ represented by their face-defining set.

We define the combinatorial automorphism group $\text{Comb}(\mathcal{C})$ of $\mathcal{C}$ as the largest subgroup of $\text{Sym}_m$ that acts invariantly on each set $\mathcal{F}_k$ for $1 \leq k \leq n-1$. For a change we will work with the $\mathcal{V}$-representation and assume that $\mathcal{C} = \text{cone}(A)$ where $A$ has column $a_1, \ldots, a_m$. What follows can also be done with an $\mathcal{H}$-representation, but in that case dimension $k$ must be swapped with $n-k$.

The combinatorial automorphism group of $\mathcal{C}$ can equivalently be defined as the largest subgroup of $\text{Sym}_m$ that acts invariantly on only the set $\mathcal{F}_{n-1}$, i.e. the facets. Because any other face can be written as an intersection of facets. Furthermore let $\text{Skel}_k(\mathcal{C})$ be the largest subgroup of $\text{Sym}_m$ that acts invariantly on each set $\mathcal{F}_l$ for $1 \leq l \leq k$. Again it is enough to only act invariantly on the set $\mathcal{F}_k$. Note that $\text{Skel}_1(\mathcal{C}) = \text{Sym}_m$, $\text{Skel}_{n-1}(\mathcal{C}) = \text{Comb}(\mathcal{C})$ and $\text{Skel}_k(\mathcal{C}) \supseteq \text{Skel}_{k+1}(\mathcal{C})$ for all $k \in [n-2]$.

We also define the linear automorphism group $\text{Lin}_A(\mathcal{C})$ of $\mathcal{C}$ that consists of all $\sigma \in \text{Sym}_m$ such that there exists a matrix $B \in \text{GL}_n(\mathbb{R})$, with $Ba_i = a_{\sigma(i)}$ for all $i \in [m]$. Note that $\text{Lin}_A(\mathcal{C})$ depends directly on $A$, so individual scaling of the $a_i$ can result in a different linear automorphism group. We have the following chain of subgroups:

$$\text{Lin}_A(\mathcal{C}) \subseteq \text{Comb}(\mathcal{C}) \subseteq \text{Skel}_k(\mathcal{C})$$

for any $1 \leq k \leq n-1$. Methods to obtain the above automorphism groups are treated in detail by Bremner, Dutour Sikirić, Pasechnik, Rehn and Schürmann [BDSP+14]. The linear automorphism group can easily be derived from the automorphism group of a specially constructed edge-labelled graph with $m$ nodes. A recent breakthrough of Babai [Bab16] shows that obtaining the automorphism group of a graph can be done in quasi-polynomial time. More importantly for us, algorithms exist to determine the automorphism group of such a graph efficiently in practice. To compute $\text{Skel}_k(\mathcal{C})$ the only general method known is to first obtain $\mathcal{F}_k$ and then reduce it to the problem of determining the automorphism group of a specially constructed edge-labelled bipartite graph consisting of $m + |\mathcal{F}_k|$ nodes. In particular, to compute $\text{Comb}(\mathcal{C}) = \text{Skel}_{n-1}(\mathcal{C})$, we need to know $\mathcal{F}_{n-1}$, i.e. all the facets of $\mathcal{C}$. If we are lucky we get $\text{Lin}_A(\mathcal{C}) = \text{Skel}_k(\mathcal{C})$ for a low $k \geq 1$ such that we must also have $\text{Comb}(\mathcal{C}) = \text{Lin}_A(\mathcal{C})$ by inclusion. However, in general it is often too hard to obtain the full combinatorial automorphism group $\text{Comb}(\mathcal{C})$.

So in practice we have to use the possibly smaller linear automorphism group $\mathrm{Lin}_A(\mathcal{C})$, which can efficiently be computed. Often this subgroup is close to the full combinatorial automorphism group $\mathrm{Comb}(\mathcal{C})$ or it is large enough for our goal.

## 3. An upper bound on the number of perfect forms

In this section we prove a both asymptotic and absolute upper bound on the number of non-similar $d$-dimensional perfect forms. Our bound of $e^{O(d^2 \log(d))}$ improves on the bound $e^{O(d^3 \log(d))}$ proven by Bacher in the preprint [Bac17]. Bacher already conjectured such an upper bound with heuristic arguments. Our proof strategy does not seem to overlap with the proof or the heuristics of Bacher.

**Theorem 3.** *The number $p_d$ of non-similar $d$-dimensional perfect quadratic forms $Q$ is upper bounded by $e^{O(d^2 \log(d))}$.*

The proof makes use of a volumetric argument and dual HKZ reduction. An important part of the proof is that the Voronoi domains of perfect forms with fixed arithmetical minimum form an essentially disjoint packing of the rational closure $\tilde{\mathcal{S}}^d_{\geq 0} = \text{cone}\{xx^t : x \in \mathbb{Z}^n\}$.

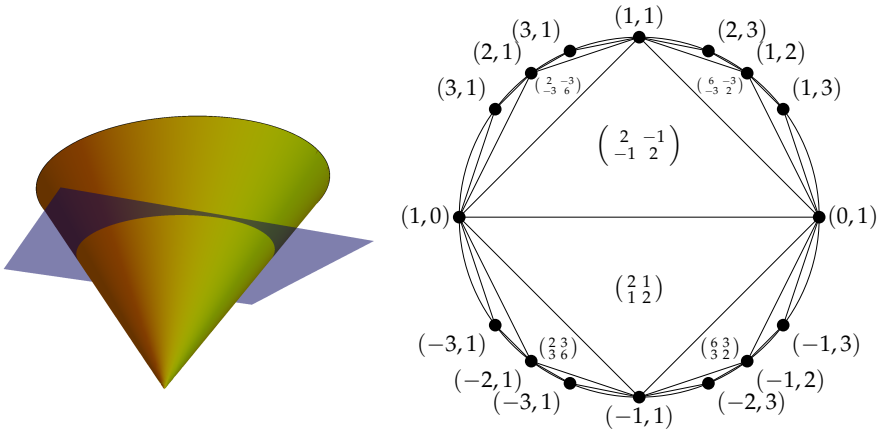


FIGURE 6. Subdivision by Voronoi domains in dimension 2. A vector $x$ indicates the extreme ray $xx^t$. A matrix $Q$ indicates the Voronoi domain $\mathcal{V}(Q)$.

**Lemma 4.** *(Voronoi [Voro8]) The Voronoi domains of the $d$-dimensional perfect forms $Q$ form a polyhedral subdivision of $\tilde{\mathcal{S}}^d_{\geq 0}$. In particular, if we restrict ourselves to perfect forms with $\lambda(Q) = 1$, we get that*

$$\tilde{\mathcal{S}}^d_{\geq 0} = \bigcup_{\substack{Q \text{ perfect} \\ \lambda(Q)=1}} \mathcal{V}(Q),$$

*where the intersection of the interior of any two Voronoi domains, from distinct perfect forms, is empty.*

*Proof.* This result originates from a reduction theory of Voronoi [Voro8], see section 7.1 of [Maro2] for a full proof. We do reprove the last part of the Lemma, that is, the part that the union is essentially disjoint. This is the only important part from this Lemma that we need. Let $Q, Q' \in \mathcal{S}^d_{>0}$ be two perfect forms where we assume that $\lambda(Q) = \lambda(Q') = 1$. Suppose that there exists an $R \in \text{Int}(\mathcal{V}(Q)) \cap \text{Int}(V(Q'))$. We show that then $Q = Q'$. Because $R \in \text{Int}(\mathcal{V}(Q))$, there exist $c_i > 0$ such that $R = \sum_{x_i \in \text{Min}\, Q} c_i x_i x_i^t$. Then we get that

$$\langle R, Q' \rangle = \sum_{x_i \in \text{Min}\, Q} c_i x_i^t Q' x_i \geq \sum_{x_i \in \text{Min}\, Q} c_i = \sum_{x_i \in \text{Min}\, Q} c_i x_i^t Q x_i = \langle R, Q \rangle,$$

using that $\lambda(Q) = \lambda(Q') = 1$. By $R \in \mathrm{Int}(V(Q'))$ we get analogous the inequality $\langle R, Q' \rangle \leq \langle R, Q \rangle$ and thus equality. Then we have

$$0 = \langle R, Q' - Q \rangle = \sum_{x_i \in \mathrm{Min}\, Q} c_i \left( x_i^t Q' x_i - 1 \right).$$

Because $c_i > 0$ and $x_i^t Q' x_i \geq 1$ for all $i$, we must have that $x_i^t Q' x_i = 1$ for all $i$, i.e. $\mathrm{Min}\, Q \subset \mathrm{Min}\, Q'$. We conclude by perfectness of $Q$ that $Q' = Q$. □

To turn Lemma 4 into an upper bound on the number of non-similar perfect forms, we need in every similarity class a perfect form $Q$ for which $\mathcal{V}(Q)$ is 'large'. First we need to the following Lemma which makes use of dual HKZ reduction.

**Lemma 5.** *Consider a PQF $Q \in \mathcal{S}^d_{>0}$. Then there exists a $Q' \in \mathcal{S}^d_{>0}$ arithmetically equivalent to $Q$ such that $x^t x \leq \frac{1}{8} d^3 (d+7)$ for all $x \in$ Min $Q'$.*

*Proof.* Remind yourself that two forms are arithmetically equivalent if they are $\mathrm{GL}_d(\mathbb{Z})$-equivalent. Positive scaling has no influence on the minimal vectors and thus we assume without loss of generality that $\lambda(Q) = 1$. We also assume that the dual $Q^{-1}$ is HKZ reduced, as every equivalence class contains at least one such $Q$. Because $Q^{-1}$ is HKZ reduced, we have by Lemma 1 that

$$(Q^{-1})_{ii} \leq \frac{i+3}{4} \lambda_i(Q^{-1})$$

for all $i = [d]$. Furthermore, note that $\lambda_d(Q) \geq \ldots \geq \lambda_1(Q) = 1$. Thus by the transference theorem of Banaszczyk [Ban93] we have

$$\lambda_i(Q^{-1}) \leq \frac{d^2}{\lambda_{d-i+1}(Q)} \leq d^2.$$

As a result we obtain

$$\mathrm{Tr}(Q^{-1}) = \sum_{i=1}^{d} (Q^{-1})_{ii} \leq d^2 \cdot \sum_{i=1}^{d} \frac{i+3}{4} = \frac{1}{8} d^3 (d+7).$$

In particular, this gives a lower bound on the eigenvalues $\mu_1, \ldots, \mu_d > 0$ of $Q$, namely

$$\frac{1}{\mu_i} \leq \sum_{j=1}^{d} \frac{1}{\mu_j} = \mathrm{Tr}(Q^{-1}) \leq \frac{1}{8} d^3 (d+7).$$

But as $\min_i \mu_i = \min_{x \in \mathbb{R}^d - 0} \frac{x^t Q x}{x^t x}$ we get for all $x \in \mathrm{Min}\, Q$ that:

$$x^t x \leq \frac{x^t Q x}{\min_i \mu_i} \leq 1 \cdot \frac{1}{8} d^3 (d+7).$$

□

*Proof of Theorem 3.* Let $P_d$ be a complete set of representatives of perfect $d$-dimensional quadratic forms with $\lambda_1(Q) = 1$ up to arithmetical equivalence. To quantify the volume of a cone we bound it by a half-space. We use the half-space $T_d = \{Q \in \mathcal{S}^d : \langle Q, I_d \rangle = \mathrm{Tr}(Q) \leq 1\}$ in $\mathcal{S}^d$. By Lemma 1 we have

$$\bigcup_{Q \in P_d} \mathcal{V}(Q) \subset \tilde{\mathcal{S}}^d_{\geq 0} \subset \mathcal{S}^d_{\geq 0},$$

where the $\mathcal{V}(Q)$ share no interior. Recall the isometry $\phi : \mathcal{S}^d \to \mathbb{R}^n$ from Section 2.2. This yields

$$\sum_{Q \in P_d} \mathrm{Vol}(\phi(\mathcal{V}(Q) \cap T_d)) \leq \mathrm{Vol}(\phi(\tilde{\mathcal{S}}^d_{\geq 0} \cap T_d)) \leq \mathrm{Vol}(\phi(\mathcal{S}^d_{\geq 0} \cap T_d)).$$

The proof can be summarized as follows: by Lemma 5 we can assume $P_d$ consists of representatives such that the $n$-dimensional volume of $\phi(\mathcal{V}(Q) \cap T_d)$ is lower bounded by some $\ell_d$. Furthermore we can upper bound the $n$-dimensional volume of $\phi(\mathcal{S}^d_{\geq 0} \cap T_d)$ by some $u_d$. As a result we get by the subdivision $p_d = |P_d| \leq u_d / \ell_d$.

We first obtain an easy upper bound on the volume of $\phi(\mathcal{S}^d_{\geq 0} \cap T_d)$. Let $A = (a_{ij})_{i,j} \in \mathcal{S}_{\geq 0} \cap T_d$. Because $A$ is positive semidefinite, we have $a_{ii} \geq 0$ for all $i \in [d]$. Furthermore the determinant of every $2 \times 2$ minor of $A$ must be non-negative, i.e. $a_{ii}a_{jj} \geq a_{ij}^2$ for all $i, j \in [d]$, using that $a_{ij} = a_{ji}$. Because $A \in T_d$, we also get

$$1^2 \geq \left( \sum_{i=1}^{d} a_{ii} \right)^2 = \sum_{i,j} a_{ii}a_{jj} \geq \sum_{i,j} a_{ij}^2 = \langle A, A \rangle = \|\phi(A)\|^2,$$

and thus $\phi(A) \in B_1^n$, the ball of radius 1. As a result the volume of $\phi(\mathcal{S}^d_{\geq 0} \cap T_d)$ is bounded by the volume of an $n$-dimensional unit ball. This gives us the upper bound

$$\mathrm{Vol}(\phi(\mathcal{S}^d_{\geq 0} \cap T_d)) \leq \frac{\pi^{n/2}}{\Gamma(n/2+1)} =: u_d.$$

Note that $\Gamma(n/2+1)$ eventually grows much faster than $\pi^{n/2}$. This results in the volume of $\phi(\mathcal{S}^d_{\geq 0} \cap T_d)$ going to 0 as $d \to \infty$. So for $d$ large enough we can assume that $u_d \leq 1$.

Now we lower bound the volume of $V' = \phi(\mathcal{V}(Q) \cap T_d)$ for a well chosen representative $Q$ of an equivalence class of perfect forms with $\lambda_1(Q) = 1$. Then the polytope $V'$ is the convex hull of 0 and the embedding of $\frac{xx^t}{x^t x}$ for all $x \in \mathrm{Min}\, Q / \{\pm 1\}$. This because the trace of $xx^t$ is exactly given by $x^t x$. As we are only in search of a lower bound for the volume, we will consider without loss of generality a subset $M_Q \subset \mathrm{Min}\, Q$ with $|M_Q| = n$ and $\mathrm{rank}\{\phi(xx^t) \in \mathbb{R}^n : x \in M_Q\} = n$. Note that this is possible exactly because $Q$ is perfect. Denote $M_Q = \{x_1, \ldots, x_n\} \subset \mathbb{Z}^d$. Then we get

$$\mathrm{Vol}(V') \geq \mathrm{Vol}\left( \mathrm{conv}\left( \{0\} \cup \left\{ \phi\left( \frac{x_i x_i^t}{x_i^t x_i} \right) : i \in [n] \right\} \right) \right) = \frac{1}{n!} |\det(U)|$$

with

$$U = \left( \phi\left( \frac{x_i x_i^t}{x_i^t x_i} \right) \right)_{i=1,\ldots,n} \in \mathbb{R}^{n \times n}.$$

By Lemma 5 we can now assume that $x_i^t x_i \leq \frac{1}{8}d^3(d+7)$ for all $i = 1, \ldots, n$. Furthermore note that $\phi\left(\frac{x_i x_i^t}{x_i^t x_i}\right) = \frac{\phi(x_i x_i^t)}{x_i^t x_i}$. Then we get

$$\begin{aligned} |\det(U)| &= \prod_{i=1}^{n} \frac{1}{x_i^t x_i} \cdot |\det((\phi(x_i x_i^t))_{i=1,\ldots,n})| \\ &\geq \left(\prod_{i=1}^{n} \frac{1}{x_i^t x_i}\right) \cdot 2^{(n-d)/2} \geq \frac{2^{(n-d)/2}}{(\frac{1}{8}d^3(d+7))^n} \end{aligned}$$

using that $\phi(x_i x_i^t) \in \mathbb{Z}^d \oplus \sqrt{2}\mathbb{Z}^{n-d}$ and that the determinant of the matrix $(\phi(x_i x_i^t))_{i\in[n]}$ is nonzero, because it has full rank. So we can conclude that

$$\mathrm{Vol}(\phi(\mathcal{V}(Q) \cap T_d)) \geq \frac{1}{n!} \frac{2^{(n-d)/2}}{(\frac{1}{8}d^3(d+7))^n} := \ell_d$$

for at least one representative $Q$ in each equivalence class of perfect $d$-dimensional quadratic forms with $\lambda_1(Q) = 1$.

Recall that $n = \binom{d+1}{2} = O(d^2)$ and $n! \leq n^n = e^{O(d^2 \log(d))}$. In conclusion, we get that

$$p_d \leq \frac{u_d}{\ell_d} \leq \frac{1}{\frac{1}{n!}\frac{2^{(n-d)/2}}{(\frac{1}{8}d^3(d+7))^n}} = n! \cdot 2^{-(n-d)/2} \cdot (\frac{1}{8}d^3(d+7))^n = e^{O(d^2 \log(d))}.$$

An absolute bound is given by:

$$p_d \leq \frac{u_d}{\ell_d} = \frac{\pi^{n/2}}{\Gamma(n/2+1)} \cdot n! \cdot 2^{-(n-d)/2} \cdot (\frac{1}{8}d^3(d+7))^n.$$

□

## 4. Voronoi's algorithm

In Voronoi's celebrated paper [Voro8] he introduced an algorithm to find all non-similar perfect forms. Even though the correctness proofs of Voronoi's algorithm have been simplified in the past century, the original algorithm remains the most efficient general algorithm for classifying perfect forms known so far, that is proven to be exhaustive. Even earlier in 1873, Korkine and Zolotarev [KZ73] classified all perfect forms up to dimension 5. Voronoi was able to run his algorithm by hand to verify their results. Furthermore, by executing his algorithm partially, Voronoi conjectured that the list of seven 6-dimensional perfect forms he found was complete. Only in 1957, in a lengthy paper full of calculations, Barnes [Bar57] succeeded in finishing Voronoi's algorithm in dimension 6 and thereby proving that Voronoi's list was complete.

In 1963 Scott [Sco63] tried to apply the methods of Barnes to dimension 7, but he succeeded only partially and there were some errors in his results. In 1975, using a theorem of Watson [Wat69], Stacey [Sta75] obtained an exhaustive list of thousands of 7-dimensional forms. Using simple invariants he extracted a not necessarily complete list of 33 non-similar forms. The complexity of Voronoi's algorithm and the number of perfect forms rise quickly when the dimension increases. Therefore, it was only in 1993 that Jaquet [JC93], using computer assistance, was able to fully run Voronoi's algorithm in dimension 7 and thereby proving the completeness of Stacey's list. He needed more than 4 months of CPU time to achieve this on a VAX 8530.

After constructive work by Laïhem [Laï92] and Baril [Bar96], a total of $1175 + 53$ 8-dimensional perfect forms were known. By running Voronoi's algorithm partially Napias [Nap96] found a total of 10770 perfect 8-dimensional forms, which was extended to 10916 by Batut and Martinet [BMoo] in 2000. At the time it was deemed almost impossible to finish Voronoi's algorithm for dimension 8. However, in 2005, Dutour Sikirić, Schürmann and Vallentin [DSSV07] were finally able to finish the computation using more than 15 months of computing time (hardware not mentioned). They proved that there are 10916 perfect non-similar 8-dimensional forms. They also found more than 500.000 perfect 9-dimensional forms, by running Voronoi's algorithm partially, with no end in sight.

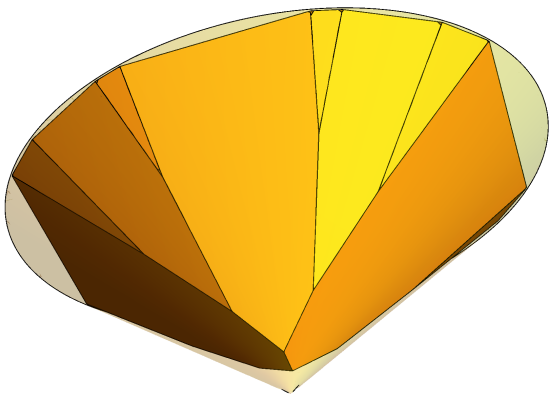

Figure 7. Ryshkov Polyhedron $\mathcal{P}_1$ inside of $\mathcal{S}^2_{\geq 0}$.

4.1. **Ryshkov Polyhedra.** We define for a fixed $\lambda > 0$ the Ryshkov Polyhedron [Rys70]

$$\mathcal{P}_\lambda = \{Q \in \mathcal{S}^d_{>0} : \lambda(Q) \geq \lambda\},$$

or equivalently as an intersection of infinitely many half-spaces

$$\mathcal{P}_\lambda = \bigcap_{x \in \mathbb{Z}^d \setminus \{0\}} \{Q \in \mathcal{S}^d : Q[x] = \langle Q, xx^t \rangle \geq \lambda\} \subset \mathcal{S}^d_{>0}.$$

Even though $\mathcal{P}_\lambda$ is given by the intersection of infinitely many half-spaces, it is a locally finite polyhedron. In our setting this means that if we intersect $\mathcal{P}_\lambda$ with any half-space such that the resulting polyhedron $\mathcal{P}'$ is bounded, then $\mathcal{P}'$ is given by an intersection of only a finite number of half-spaces. See Theorem 3.1 by Schürmann [Sch09] for a proof. By construction the Hermite constant can now be be expressed as

$$\mathcal{H}_d = \frac{\lambda}{\inf\limits_{Q \in \mathcal{P}_\lambda} (\det(Q)^{1/d})}.$$

In combination with the above, the importance of perfect forms follows immediately from Theorem 6.

**Theorem 6.** *(Minkowski) The function $Q \mapsto (\det Q)^{1/d}$ is concave on $\mathcal{S}^d_{>0}$. It is even strictly concave, except between two forms that are a scalar multiple of each other.*

*Proof.* Let $A, B \in S^d_{>0}$. The proof follows from the famous Minkowski determinant inequality for positive definite matrices

$$\det(A+B)^{1/d} \geq \det(A)^{1/d} + \det(B)^{1/d},$$

which is strict in case $A$ is not a scalar multiple of $B$. For an elementary proof note that if $B = CC^t$ is the Cholesky decomposition of $B$, then it is equivalent to prove the above inequality for $A' = C^{-1}AC^{-t}$ and $B = I$, i.e., to show that $\det(A'+I)^{1/d} \geq \det(A')^{1/d} + 1$. Let $\mu_1, \ldots, \mu_d > 0$ be the eigenvalues of $A'$. Then, by the arithmetic and geometric mean (AM-GM) inequality, we have that

$$\begin{aligned}\det(A'+I) = \prod_{i=1}^{d}(\mu_i + 1) = \sum_{j=0}^{d} \sum_{\substack{S \subset [d] \\ |S|=j}} \prod_{j \in S} \mu_j \geq \sum_{j=0}^{d} \binom{d}{j} \left( \prod_{\substack{S \subset [d] \\ |S|=j}} \prod_{j \in S} \mu_j \right)^{1/\binom{d}{j}} \\ = \sum_{j=0}^{d} \binom{d}{j} \left( \prod_{i=1}^{d} \mu_j \right)^{j/d} = \left( \left( \prod_{i=1}^{d} \mu_i \right)^{1/d} + 1 \right)^d = (\det(A')^{1/d} + 1)^d.\end{aligned}$$

Note that if $A$ is not a scalar multiple of $B$, then the eigenvalues $\mu_1, \ldots, \mu_d$ of $A'$ are not all identical and thus the AM-GM inequality is strict. □

If we want to minimize a strictly concave function over a convex set it is well known that the local and global optimal values must lie at the boundary and in our case the vertices. To get an intuition note that for any point $a$ in the set not on the boundary we can pick two other points $b$ and $c$ in the set, with $a$ on a line between these points. Because the function is strictly concave it must take a strictly lower value at $b$ or $c$ than at $a$. If our convex set is a Ryskov Polyhedron we can also find such two points for any point that is not a vertex. So by Theorem

6 local maxima of the Hermite invariant $\gamma$ on $\mathcal{S}^d_{>0}$ can only be attained on the vertices of some Ryshkov Polyhedron $\mathcal{P}_\lambda$.

As the Hermite invariant is invariant under scaling the global optima $\mathcal{H}_d$ of $\gamma$ must be attained by at least one vertex of every Ryshkov Polyhedron $\mathcal{P}_\lambda$. Note that the vertices of $\mathcal{P}_\lambda$ are exactly the perfect forms $Q \in \mathcal{S}^d_{>0}$ with $\lambda(Q) = \lambda$, because these vertices are uniquely defined by the intersection of the hyperplanes belonging to the minimal vectors. As a result, a complete classification of the perfect forms in dimension $d$, is enough to determine the Hermite constant $\mathcal{H}_d$, hence to solve the lattice packing problem in that dimension. This is equivalent to a complete classification of the vertices of $\mathcal{P}_\lambda$ for a fixed $\lambda > 0$ under the action of $\mathrm{GL}_d(\mathbb{Z})$. For simplicity we assume from now on that $\lambda = 1$.

In particular extreme forms, i.e. local maxima of the Hermite invariant, are always perfect. For a perfect form to be extreme, Voronoi showed that it must be eutactic.

**Theorem 7.** *(Voronoi [Voro8]) A PQF $Q \in \mathcal{S}^d_{>0}$ is extreme if and only if $Q$ is perfect and eutactic.*

*Proof.* We give a geometric proof by A. Schürmann [Scho9]. A simple computation shows that the gradient of the function

$$\begin{aligned} \det : \mathcal{S}^d &\to \mathbb{R}, \\ Q &\mapsto \det(Q) \end{aligned}$$

satisfies

$$\operatorname{grad} \det Q = (\det Q) Q^{-1} \in \mathcal{S}^d.$$

Fix an arbitrary perfect $Q \in \mathcal{S}^d_{>0}$. Consider the smooth fixed determinant surface

$$S = \{Q' \in S^d_{>0} : \det Q' = \det Q\}$$

and its tangent hyperplane in $Q$ given by

$$T = \{Q' \in \mathcal{S}^d : \langle Q^{-1}, Q' \rangle = \langle Q^{-1}, Q \rangle\}.$$

By concavity of the determinant function, see Theorem 6, $S$ is contained in the half-space

$$H = \{Q' \in \mathcal{S}^d : \langle Q^{-1}, Q' \rangle \geq \langle Q^{-1}, Q \rangle\}$$

with boundary $T$ and with $Q$ the unique intersection point of $S$ and $T$. So a perfect form $Q$ attains a local minimum of $\det Q$ if and only if $\mathcal{P}_{\lambda(Q)}$ is fully contained in $H$ and such that $H \cap \mathcal{P}_{\lambda(Q)} = \{Q\}$. But this is the case if and only if $Q^{-1}$ lies in the interior of the inner normal cone $\mathcal{V}(Q) = \operatorname{cone}(\{xx^t : x \in \operatorname{Min} Q\})$ of $\mathcal{P}_{\lambda(Q)}$ at $Q$. □

4.2. **Voronoi's algorithm.** Starting with one perfect form Voronoi's algorithm determines all perfect forms up to similarity. More detailed we determine all vertices of the Ryskov Polyhedron $\mathcal{P}_1$, which correspond to perfect forms with arithmetical minimum 1, up to arithmetical equivalence. Voronoi's algorithm achieves this by examining all neighbouring vertices of already known vertices. An overview of Voronoi's algorithm is shown as Algorithm 1.

**Input:** Dimension d.
**Output:** A complete list of non-similar perfect forms in $\mathcal{S}^d_{>0}$.

(1) Start with a perfect form $Q$ such that $\lambda(Q) = 1$.
(2) Compute $\operatorname{Min} Q$.
(3) Enumerate the extreme rays $R_1, \ldots, R_k$ of the cone

$$\mathcal{P}(Q) = \{Q' \in \mathcal{S}^d : \langle Q', xx^t \rangle \geq 0 \text{ for all } x \in \operatorname{Min} Q\}.$$

(4) Determine contiguous perfect forms $Q_i = Q + \alpha R_i$ for $i = 1, \ldots, k$.
(5) Test if $Q_i$ is arithmetically equivalent to an already known form.
(6) Repeat steps $(2) - (5)$ for each new form.

**Algorithm 1:** Voronoi's algorithm

$\mathcal{P}(Q)$, as defined in Algorithm 1, is exactly the tangent cone of $\mathcal{P}_1$ at the perfect form $Q \in \mathcal{P}_1$ with $\lambda(Q) = 1$. As a result the extreme rays $R_i$ correspond exactly to the edges of $\mathcal{P}_1$ from the vertex $Q$. All vertices of $\mathcal{P}_1$ are connected by some path of edges, so all perfect forms are connected and thus the resulting list of perfect forms is complete if the algorithm terminates. Because there are only a finite number of perfect forms up to similarity Voronoi's algorithm terminates.

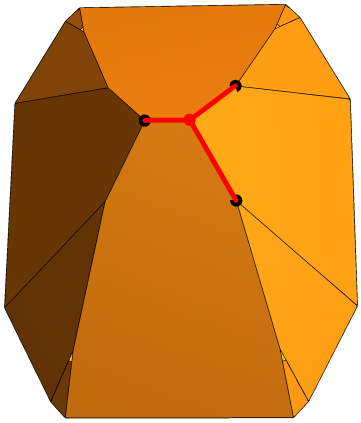

FIGURE 8. Voronoi's algorithm in dimension 2.

**Example 8.** This example is also illustrated in Figure 8. We start Voronoi's algorithm in dimension 2 with the perfect form

$$Q = \begin{pmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{pmatrix},$$

which has minimal vectors $\pm(1,0), \pm(0,1)$ and $\pm(1,1)$. Then the tangent cone $\mathcal{P}(Q)$ is given by the inequalities

$$\left\langle Q', \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \right\rangle \geq 0, \left\langle Q', \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \right\rangle \geq 0 \text{ and } \left\langle Q', \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \right\rangle \geq 0$$

and has extreme rays in the directions

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 2 & -1 \\ -1 & 0 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & -1 \\ -1 & 2 \end{pmatrix}.$$

These extreme rays lead to the 3 contiguous perfect forms

$$\begin{pmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & 1 \end{pmatrix}, \begin{pmatrix} 3 & -\frac{3}{2} \\ -\frac{3}{2} & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 3 \end{pmatrix}.$$

All these perfect forms are arithmetically equivalent to $Q$, so there exists just one 2-dimensional perfect form.

4.3. **Implementation details.** We give here a short overview of the implementation of Voronoi's algorithm. Most details are given in the later sections. To start the algorithm, we must have at least one perfect form. For this we take a quadratic form associated to the root lattice $A_d$, which is perfect for all $d$. Take for example $Q_{A_d}$ with 1 on the diagonal and $\frac{1}{2}$ on the band directly above and below the diagonal.

In step (2) and, as we see later also in steps (4) and (5), we must be able to compute $\operatorname{Min} Q$. For our approach of steps (4) and (5), we need to compute for any $C > 0$ all $x \in \mathbb{Z}^d \setminus \{0\}$ such that $x^t Q x \leq C$. The enumeration algorithm by Fincke and Pohst [FP85] allows us to enumerate all such $x$. Despite not being the most efficient algorithm asymptotically, it is very fast in the low dimensions we work with. The algorithm is based on the Lagrange expansion of $Q$, which implies

$$\left| x_i - \sum_{i=1}^{d} \alpha_{ij} x_j \right| \leq \sqrt{\frac{C}{A_i}}.$$

As a result the possible values of $x_i$ are bounded in terms of $x_{i+1}, \ldots, x_d$, which, inductively, also have only a finite number of possibilities. Fincke and Pohst first apply LLL-reduction to $Q$ yielding a relatively small enumeration space.

Step (3), in terms of cones, is the problem of converting the $\mathcal{H}$-representation of $\mathcal{P}(Q)$ to its corresponding $\mathcal{V}$-representation. This problem is also known as the representation conversion or the dual description problem. Note that this is equivalent to converting the $\mathcal{V}$-representation of the dual cone $\mathcal{V}(Q) = \mathcal{P}(Q)^*$ to its corresponding $\mathcal{H}$-representation. At least until dimension 8, this step was responsible for the slow progress in running Voronoi's algorithm completely. If the cone $\mathcal{P}(Q)$ is degenerate, i.e. the number of inequalities that define it is larger than the dimension of the space, the representation conversion problem can be hard. This is mostly because the output complexity, i.e. the number of extreme rays, becomes huge. As a result any algorithm enumerating all extreme rays becomes infeasible. In dimensions $d = 6, 7, 8$ the hardest representation conversion problems were those of the cones $\mathcal{P}(Q_{E_d})$ of the PQFs corresponding to the optimal root lattices $E_d$. Luckily these cases also have many symmetries, which were heavily used by all authors that were able to complete Voronoi's algorithm in these dimensions. See Section 8 for an extensive overview of this problem, including how to make use of the symmetry.

For step (4), given an extreme ray $R$ of $\mathcal{P}(Q)$, we have to determine $\alpha > 0$ such that $Q_\alpha := Q + \alpha R$ is perfect and $\lambda(Q_\alpha) = \lambda(Q) = 1$. For $\rho > \alpha$ the quadratic form $Q_\rho$ lies outside of $\mathcal{P}_1$. So either $Q_\rho \notin \mathcal{S}^d_{>0}$ or $\lambda(Q_\rho) < 1$. For $\rho < \alpha$ the quadratic form $Q_\rho$ lies on the edge strictly between $Q$ and $Q_\alpha$ and thus in particular $\operatorname{Min} Q_\rho \subsetneq \operatorname{Min} Q$. As we can compute all these properties, it is easy to obtain a binary search algorithm that converges to the correct $\alpha$. In Section 5 we go into detail on how to optimize this step of the algorithm, both asymptotically and in practice.

Step (5) is the Lattice Isomorphism Problem (LIP). Given $Q, Q' \in \mathcal{S}^d_{>0}$ we need to determine if there exists a $U \in \mathrm{GL}_d(\mathbb{Z})$ such that $Q' = U^t Q U$. The existence of such a $U$ implies that $q'_{ii} = Q'[e_i] = Q[u_i]$ and thus we have only finitely many choices for $u_i$. This idea can be enhanced further. We also show an alternative way to solve this problem, by efficiently finding a canonical representative for each perfect form, an idea recently conceived by Dutour Sikirić [DS18]. See Section 6 for more details.

## 5. Determining contiguous perfect forms

We present two algorithms in this section, the first one is fast in practice for low dimensions and the second is fast asymptotically. Before describing these algorithms in detail we introduce the general framework on which they are based. Suppose $Q \in \mathcal{S}^d_{>0}$ is a perfect form with $\lambda(Q) = 1$ and $R$ an extreme ray of $\mathcal{P}(Q)$. We want to determine the unique $\alpha > 0$ such that $Q_\alpha := Q + \alpha R$ is perfect and $\lambda(Q_\alpha) = 1$. Let $\beta > 0$ be minimal such that $Q_\beta$ is not positive definite, i.e. such that $Q_\beta$ lies on the boundary of $\mathcal{S}^d_{\geq 0}$. We make a plot of $\lambda(Q_\rho)$ for $\rho \in [0, \beta]$, which could for example look like Figure 9.

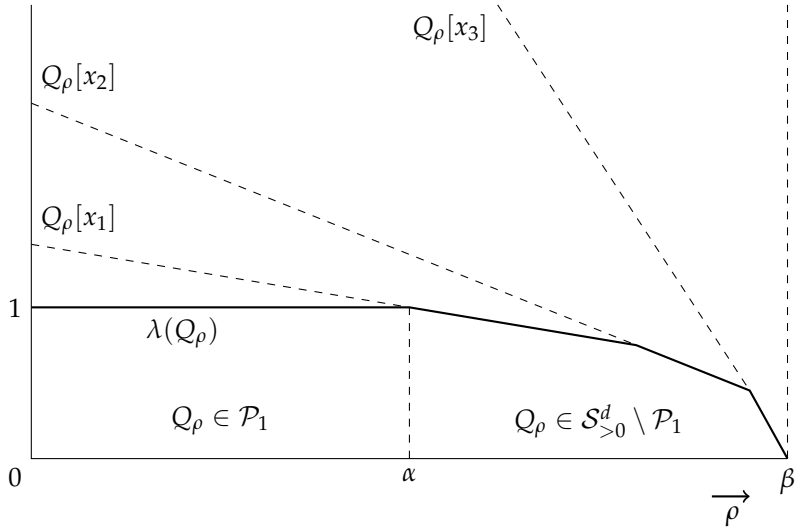


Figure 9. Example plot of $\lambda(Q_\rho)$.

Note that for $0 < \rho < \alpha$ the quadratic form $Q_\rho$ lies strictly on the edge of the Ryshkov Polyhedron $\mathcal{P}_1$ between the perfect forms $Q$ and $Q_\alpha$. For $\alpha < \rho \leq \beta$ the quadratic form $Q_\rho$ lies outside of the Ryshkov Polyhedron $\mathcal{P}_1$, i.e. it has $\lambda(Q_\rho) < 1$. When $\rho > \beta$ the quadratic form $Q_\rho$ even lies outside of $\mathcal{S}^d_{\geq 0}$.

Suppose $\rho \in [\alpha, \beta)$, then any short vector $x \in \mathbb{Z}^n \setminus \{0\}$ such that $x^t Q_\rho x < 1$, brings $\rho$ closer to $\alpha$. Namely we must have $x^t Q_\alpha x \geq 1$ and thus we can decrease $\rho$ to $\rho'$ to the point where $x^t Q_{\rho'} x = 1$. This is possible because necessarily $x^t R x < 0$. A quick calculation shows that $\rho' = (1 - x^t Q x)/(x^t R x) < \rho$. As a matter of fact, any $x \in \operatorname{Min} Q_\alpha \setminus \operatorname{Min} Q$ necessarily has $x^t Q_\rho x < 1$ for all $\rho > \alpha$. So we can repeat the above until we have $\rho = \alpha$, which at that point is detected by $\lambda(Q_\rho) = 1$. We call this the iterative method.

Let $\rho \in [\alpha, \beta)$ and consider the set $S_{<1} = \{x \in \mathbb{Z}^n \setminus \{0\} : x^t Q_\rho x < 1\}$. If $S_{<1}$ is empty we necessarily have $\alpha = \rho$. Otherwise each vector $x \in S_{<1}$ gives an upper bound $(1 - x^t Q x)/(x^t R x)$ on $\alpha$ where the least upper bound is sharp, i.e.

$$\alpha = \min_{x \in S_{<1}} \{(1 - x^t Q x)/(x^t R x)\}.$$

To see that this must be sharp note that for $x \in \operatorname{Min} Q_\alpha \setminus \operatorname{Min} Q$ we necessarily have $x^t R x < 0$. As a result $x^t Q_\rho x < 1$ for all $\rho > \alpha$ and thus $x \in S_{<1}$. We call this the direct method. The problem is that $\lambda(Q_\rho)$ can be very small and as a result $S_{<1}$, which grows by a factor $(1/\lambda(Q_\rho))^{O(d)}$, can become very large.

Enumerating the full set $S_{<1}$ in this case is expensive and therefore the iterative method mentioned earlier is more efficient as long as $\lambda(Q_\rho)$ is small.

It remains to obtain such a $\rho \in [\alpha, \beta)$, without any knowledge of $\alpha$. But first we consider how to obtain $\beta$. As $\beta > 0$ is minimal such that $Q_\beta$ is not positive definite, we have in particular that $\beta > 0$ is minimal such that $\det(Q_\beta) = 0$. Let $Q = CC^T$ be the Cholesky decomposition of $Q$ and note that $\det(Q + \alpha R) = 0$ if and only if $\det(I + \alpha R') = 0$ with $R' := C^{-1}R(C^{-1})^T$. Furthermore $R'$ is symmetric and if it has eigenvalues $\mu_1 \leq \ldots \leq \mu_d \in \mathbb{R}$, then $\det(I + xR') = 0$ if and only if $x = -\frac{1}{\mu_i}$ for some $i$. We find that $\beta = -\frac{1}{\mu_1} > 0$ using that $R'$ has at least one negative eigenvalue, because it is not positive semidefinite. We could also have used $R' = Q^{-1}R$, but then we lose a lot of numerical stability and efficiency in obtaining the eigenvalues, because $R'$ is not symmetric any more. The complexity of this computation is negligible compared to the whole algorithm of determining a contiguous perfect form. Also we assume that $\beta$ can be obtained up to the necessary precision. Another option would be to binary search for $\beta$ using the property of being positive definite.

**Lemma 9.** *Suppose $Q \in \mathcal{S}^d_{>0}$ is perfect and $R$ an extreme ray of $\mathcal{P}(Q)$. Let $\alpha > 0$ be such that $Q + \alpha R$ is perfect and let $\beta > \alpha$ be minimal such that $Q + \beta R$ is not positive definite. Let $Q = CC^t$ be the Cholesky decomposition of $Q$ and let $R' = C^{-1}RC^{-t}$. Suppose $R'$ has eigenvalues $\mu_1 \leq \cdots \leq \mu_d$. Even if $\alpha$ is unknown we can find in polynomial time some $\rho \in (\alpha, \beta)$ such that*

$$\beta - \rho \geq \frac{1}{2}\left(\det(Q) \cdot h_d^d \cdot \prod_{i=2}^{d}\left(1 + \frac{|\mu_i|}{|\mu_1|}\right)\right)^{-1} \cdot \beta,$$

*where $h_d$ is any upper bound for the Hermite constant $\mathcal{H}_d$.*

*Proof.* Note that $\lambda(Q_\rho) = 1$ for all $0 \leq \rho \leq \alpha$, so in order to get $\rho > \alpha$ we only need to find a $\rho \in (0, \beta = -\frac{1}{\mu_1})$ such that $\lambda(Q_\rho) < 1$. If $\det(Q_\rho)$ is small enough this follows from any bound on the Hermite constant, because $\lambda(Q) \leq \det(Q)^{1/d} \cdot \mathcal{H}_d$ by definition. Because $\det(Q_\beta) = 0$, we can move to $Q_{\beta-\varepsilon}$ for small $\varepsilon > 0$, without increasing the determinant too much. To make this more precise, note that

$$\det(Q_\rho) = \det(Q) \cdot \det(I + \rho R') = \det(Q) \cdot \prod_{i=1}^{d}(1 + \rho\mu_i).$$

In order to obtain $\lambda(Q_\rho) < 1$, we need $\det(Q_\rho) < h_d^{-d} \leq \mathcal{H}_d^{-d}$ for any upper bound $h_d$ of the Hermite constant $\mathcal{H}_d$. Now, using that $\rho \in (0, -1/\mu_1)$, we get

$$\begin{aligned}\det(Q_\rho) = \det(Q) \cdot \prod_{i=1}^{d}(1 + \rho\mu_i) &= \det(Q) \cdot (1 + \rho\mu_1) \cdot \prod_{i=2}^{d}(1 + \rho\mu_i) \\ &\leq \det(Q) \cdot (1 + \rho\mu_1) \cdot \prod_{i=2}^{n}\left(1 + \frac{|\mu_i|}{|\mu_1|}\right).\end{aligned}$$

Recall that $\beta \cdot \mu_1 = -1$. Therefore, putting

$$\rho = \left(1 - \frac{1}{2}\left(\det(Q) \cdot h_d^d \cdot \prod_{i=2}^{d}\left(1 + \frac{|\mu_i|}{|\mu_1|}\right)\right)^{-1}\right) \cdot \beta < \beta$$

makes sure that $\det(Q_\rho) \leq \frac{1}{2}h_d^{-d} < h_d^{-d}$. This guarantees that $\lambda(Q_\rho) < 1$ and thus $\rho \in (\alpha, \beta)$. □

5.1. **A practical algorithm.** Previous implementations by Dutour Sikirić, et al and Martinet [DSSV07, Mar02] used the values $\lambda(Q_\rho)$, $\operatorname{Min} Q_\rho$ and a check for positive definiteness to apply binary search for $\alpha$. Especially computing $\lambda(Q_\rho)$ and $\operatorname{Min} Q_\rho$ is expensive. For dimensions up to 8, doing this computation efficiently was not so important, due to the low number of perfect forms. However, for dimensions 9 or higher, because of the large number of perfect forms, a more efficient implementation is needed.

As explained in the introduction of this section we can efficiently obtain a good approximation of $\beta$ from an approximation of the smallest eigenvalue of $R'$. Using Lemma 9, we can calculate a starting value $\rho_1 \in (\alpha, \beta)$. Then, in each iteration step $i \geq 1$, we need to find a vector $x \in \mathbb{Z}^n \setminus \{0\}$ such that $x^t Q_{\rho_i} x < 1$. The polynomial time LLL-algorithm cannot guarantee to find such a vector if it exists. However, in practice LLL performs much better in these low dimensions than what the theoretical bounds imply [NS06]. So we first try to find short vectors of $Q_{\rho_i}$ using LLL. LLL can give up to $d$ short vectors, so in each iteration we try to improve $\rho_i$ with at most $d$ short vectors. This is repeated until LLL fails to find an improving short vector at $Q_{\rho_k}$.

In practice, although this isn't guaranteed, we often already obtain $\alpha = \rho_k$ at the point that LLL fails to find an improving short vector. To conclude, we run an enumeration algorithm to find the set $S = \{x \in \mathbb{Z}^n \setminus \{0\} : x^t Q_{\rho_k} x \leq 1\}$. This enumeration is quite efficient, because $1/\lambda(Q_{\rho_k})$ is close to 1 in practice. Because $S_{<1} \subset S$, we can derive the value of $\alpha$, by the direct method mentioned in the introduction of this section. As a by-product, we obtain the subset $\operatorname{Min} Q_\alpha \subset S$. This subset is needed in later steps of Voronoi's algorithm. This gives us Algorithm 2.

**Input:** A perfect form $Q$ with $\lambda(Q) = 1$ and an extreme ray $R$ of $\mathcal{P}(Q)$
**Output:** A contiguous perfect form $Q + \alpha R$ with $\alpha > 0$ and also $\operatorname{Min} Q_\alpha$.

Calculate $\beta$ and $\rho_1 \in (\alpha, \beta)$ as in Lemma 9;
$i \leftarrow 1$;
**while** *LLL finds vectors* $x_1, \ldots, x_l$ *such that* $x_j^t Q_{\rho_i} x_j < 1$ **do**
    $\rho_{i+1} \leftarrow \min\{(1 - x_j^t Q x_j)/(x_j^t R x_j) : j = 1, \ldots, l\}$;
    $i \leftarrow i + 1$;
$S \leftarrow \{x \in \mathbb{Z}^n \setminus \{0\} : x^t Q_{\rho_i} x \leq 1\}$;
$S_{<1} \leftarrow \{x \in S : x^t Q_{\rho_i} x < 1\}$;
$\alpha \leftarrow \min_{x \in S_{<1}} \{(1 - x^t Q x)/(x^t R x)\}$;
$\operatorname{Min} Q_\alpha \leftarrow \{x \in S : x^t Q_\alpha x = 1\}$;

**Algorithm 2:** A practical algorithm for finding contiguous perfect forms

If we would start with a value $\rho_1 < \alpha$, then this algorithm would not change $\rho_1$ at all and the resulting $Q_{\rho_1}$ would not be perfect. If the output $Q_{\rho_1}$ is not perfect, i.e. $\rho_1 < \alpha$, we can efficiently notice this. Because then for all $x \in \operatorname{Min} Q_{\rho_1}$, which is already computed, we have $x^t R x \geq 0$. Starting with a lower value of $\rho_1$ than one that guarantees $\rho_1 > \alpha$ can reduce the number of iterations needed. Although starting with a value $\rho_1$ that is too low means that we have to restart the algorithm with a higher starting value, experiments show that this pays off in practice, after some tuning. Furthermore, when starting with a lower $\rho_1$, more experiments showed that running the LLL step just once was, on average, the

most efficient. Namely, because most of the time this single step already finds the correct $\alpha$.

5.2. **An asymptotically fast algorithm.** Although Algorithm 2 is efficient in practice, we could only hope to prove an asymptotic bound in the order of $2^{O(d^2)}$. This is the case, because, by using LLL, we can only guarantee in the last enumeration step that $1/\lambda(Q_{\rho_i}) = 2^{O(d)}$. Then $|S|$ can be as large as $(2^{O(d)})^d = 2^{O(d^2)}$. Furthermore we need to bound the number of iterations. So the speed of convergence of $\rho_i$ to $\alpha$ needs to be controlled or, otherwise stated, we must control the growth of $\beta - \rho_i$. We consider the slightly adapted Algorithm 3.

**Input:** A perfect form $Q$ with $\lambda(Q) = 1$ and an extreme ray $R$ of $\mathcal{P}(Q)$
**Output:** A contiguous perfect form $Q + \alpha R$ with $\alpha > 0$ and also $\operatorname{Min} Q_\alpha$.

Calculate $\beta$ and $\rho_1 \in (\alpha, \beta)$ as in Lemma 9;
$i \leftarrow 1$;
**while** $Q_{\rho_i}$ *has a shortest vector* $x$ *such that* $x^t Q_{\rho_i} x \leq \frac{1}{2}$ **do**
    $\rho_{i+1} \leftarrow (1 - x^t Q x)/(x^t R x)$;
    $i \leftarrow i + 1$;
$S \leftarrow \{x \in \mathbb{Z}^n \setminus \{0\} : x^t Q_{\rho_i} x \leq 1\}$;
$S_{<1} \leftarrow \{x \in S : x^t Q_{\rho_i} x < 1\}$;
$\alpha \leftarrow \min_{x \in S_{<1}} \{(1 - x^t Q x)/(x^t R x)\}$;
$\operatorname{Min} Q_\alpha \leftarrow \{x \in S : x^t Q_\alpha x = 1\}$;

**Algorithm 3:** An asymptotically fast algorithm for finding contiguous perfect forms

**Lemma 10.** *Let $Q$ be a perfect form and $R$ a ray of $\mathcal{P}(Q)$. Given a starting value $\rho_1 \in (\alpha, \beta)$, Algorithm 3 finds the contiguous perfect form $Q + \alpha R$ in*

$$\log_2(\beta/(\beta - \rho_1)) \cdot 2^{O(d)} + d^{\frac{d}{2e} + o(d)}$$

*operations over $\mathbb{Q}$. In particular combining this with Lemma 9 we get an algorithm that runs in time*

$$\max\{1, \log_2(1 + |\mu_d|/|\mu_1|)\} \cdot 2^{O(d)} + d^{\frac{d}{2e} + o(d)}$$

*where $\mu_1 \leq \ldots \leq \mu_d$ are the eigenvalues of $R'$ as defined in Lemma 9.*

*Proof.* In each while loop the dominant computation is that of finding a shortest vector of $Q_{\rho_i}$. Shortest vectors can be obtained by the deterministic Voronoi Cell algorithm by Micciancio and Voulgaris [MV13], which has complexity $2^{O(d)}$. In order to estimate the running time, we need to bound the number of iterations of the while loop. We have $x^t Q_{\rho_i} x \leq \frac{1}{2}, x^t Q_{\rho_{i+1}} x = 1$ and $x^t Q_\beta x \geq 0$. Combining these inequalities and using that $x^t R x < 0$ gives

$$(\beta - \rho_i) x^t R x \geq -\frac{1}{2} \implies -\frac{\frac{1}{2}}{x^t R x} \geq \beta - \rho_i$$

$$(\rho_i - \rho_{i+1}) x^t R x \leq -\frac{1}{2} \implies \rho_i - \rho_{i+1} \geq -\frac{\frac{1}{2}}{x^t R x} \geq \beta - \rho_i.$$

Then

$$\beta - \rho_{i+1} = (\beta - \rho_i) + (\rho_i - \rho_{i+1}) \geq 2(\beta - \rho_i),$$

i.e. the gap $\beta - \rho_i$ grows exponentially in $i$. Because $\rho_i > 0$, the number of iterations is thus bounded by $\log_2(\beta/(\beta - \rho_1))$.

What remains is the complexity of computing the set $S$ of short vectors of $Q_{\rho_i}$ up to length 1. This set can contain more than just the shortest vectors, so we can't use the deterministic Voronoi Cell algorithm. Instead we use Kannan's enumeration algorithm [HSo7], that runs in time

$$(1/\lambda(Q_{\rho_i}))^{O(d)} \cdot d^{d/(2e)+o(d)}.$$

By construction we made sure that $1/\lambda(Q_{\rho_i}) < 2$. So the time complexity of computing $S$ becomes $d^{d/(2e)+o(d)}$.

Applying Lemma 9, together with the bound $\det(Q) \leq \lambda(Q)^d = 1$ for perfect forms and the bound $\mathcal{H}_d \leq (4/3)^{(d-1)/2}$, yields the result. □

## 6. Arithmetical equivalence

An important step of Voronoi's algorithm is determining whether a perfect form $Q$ has already been found or not up to arithmetical equivalence. This is the Lattice Isomorphism Problem (LIP). Suppose that we have a list $L$ of already found perfect forms. There are two main strategies to solve LIP for perfect forms. Either we check if $Q$ is arithmetically equivalent to any of the already found perfect forms $Q' \in L$, or we determine a canonical version of the perfect form $Q$ and check directly if $Q \in L$. For the latter it is, of course, necessary that $L$ consists only of the canonical versions.

6.1. **Check for arithmetical equivalence.** For the first strategy, given two perfect forms $Q, Q' \in \mathcal{S}^d_{>0}$ with $\lambda(Q) = \lambda(Q')$, we want to determine if $Q$ and $Q'$ are arithmetically equivalent. The main algorithm we describe was conceived by Plesken and Pohst [FP85].

If $Q$ and $Q'$ are arithmetically equivalent, then there exists a $U \in \mathrm{GL}_d(\mathbb{Z})$ such that $Q = U^t Q' U$. This implies that $q_{ii} = Q[e_i] = Q'[u_i]$ and thus we only have finitely many choices for $u_i$. More generally, consider the set $S_i = \{x \in \mathbb{Z}^n : Q'[x] = q_{ii}\}$ of short vectors of $Q'$ and note that $u_i \in S_i$ for $i = 1, \ldots, d$.

We define a $k$-partial morphism from $Q$ to $Q'$ as a $k$-tuple of vectors $(u_1, \ldots, u_k)$ such that $u_i^t Q' u_j^t = e_i^t Q e_j = q_{ij}$ for all $1 \leq i, j \leq k$. Note that the $d$-partial morphisms determine exactly the $U$ such that $Q = U^t Q' U$. Given a $k$-partial morphism $(u_1, \ldots, u_k)$ with $k < d$ we can try to extend it to a $(k+1)$-partial morphism by trying each $u_{k+1} \in S_{k+1} \setminus \{u_1, \ldots, u_{k+1}\}$. The algorithm stops either by finding a $d$-partial morphism or after determining no such morphism exists.

A straightforward implementation of the above is often infeasible, as the $S_i$ can be quite large. One way to reduce the sizes of the $S_i$ is by making the $q_{ii}$ small.This can be achieved by reduction techniques applied to $Q$, such as $LLL$ or $HKZ$ reduction. Furthermore, to obtain an efficient algorithm, we need to discard a $k$-partial morphism as soon as possible if it cannot extend to a $d$-partial morphism. Several ways to discard $k$-partial morphisms early were later introduced by Plesken and Souvignier [PS97].

Note that if $Q = Q'$ the algorithm finds automorphisms of $Q$, i.e. $U \in \mathrm{GL}_d(\mathbb{Z})$ such that $U^t Q U = Q$. In order to obtain the full automorphism group $\mathrm{Aut}(Q) \subset \mathrm{GL}_d(\mathbb{Z})$ of $Q$ without completely enumerating it, one can use the concept of strong generating sets. Slightly adapting the algorithm allows to find complementing generators for the automorphism group. For more details see the paper by Plesken and Souvignier [PS97].

6.2. **Invariants under arithmetical equivalence.** The algorithm from Section 6.1, that checks if two forms are arithmetically equivalent, is a computationally expensive part of Voronoi's algorithm. If the list $L$ of already found perfect forms is large, it is costly to check for every perfect form $Q' \in L$ if $Q$ is arithmetically equivalent to $Q'$. To overcome this problem, we first need to restrict the possible candidates $Q' \in L$, by looking at simple invariants. In fact, saving $L$ using a hash table on these invariants, we can quickly obtain a restricted list of candidates. This vastly improves the running time of Voronoi's algorithm when the set of perfect forms is large.

**Lemma 11.** *The following properties of perfect forms $Q \in \mathcal{S}^d_{>0}$ with $\lambda(Q) = 1$ are invariant under the action of $\mathrm{GL}_d(\mathbb{Z})$:*

(1) $|Min\, Q|$.

(2) $\det(Q)$.

(3) $|Aut(Q)|$.

(4) *The scale of $Q$, i.e. the minimal $s \in \mathbb{Z}_{>0}$ such that $sQ$ integral.*

(5) *The multiset $\{|x^t Q y| : x, y \in Min\,(Q)\}$.*

*Proof.* Suppose that $Q' = U^t Q U$ and $\lambda(Q) = \lambda(Q') = 1$.

(1) We have $\mathrm{Min}\, Q = U \cdot \mathrm{Min}\, Q'$ and $U$ is invertible.

(2) Immediate result of $\det(U) = \pm 1$.

(3) We have $V \in \mathrm{Aut}(Q)$ if and only if $UVU^{-1} \in \mathrm{Aut}(Q')$.

(4) Let $s$ and $s'$ be the scale of $Q$ and $Q'$ respectively. Note that $Q' = U^t Q U = (U^t S U)/s$ for $S = s \cdot Q$ integral. Also $U^t$ and $U$ are integral, so $U^t S U$ is integral, so by definition $s \geq s'$. The other way around we also get $s \leq s'$ and thus $s = s'$.

(5) Note that $x \in \mathrm{Min}\, Q'$ if and only if $Ux \in \mathrm{Min}\, Q$ and observe that $x^t Q' y = x^t U^t Q U y = (Ux)^t Q (Uy)$.

□

Observe that especially the invariant (5) from Lemma 11 could be expensive for storage. However, if $\lambda(Q) = 1$, then $|x^t Q y| \leq \frac{1}{2}$ for $x, y \in \mathrm{Min}(Q)$ with $x \neq y$. Furthermore $x^t (s \cdot Q) y$, with $s$ the scale of $Q$, is integral and thus $|x^t Q y| \in \{0, 1, \ldots, \lfloor s/2 \rfloor\}$ for different $x, y \in \mathrm{Min}(Q)$. As a result invariant (5) from Lemma 11 can equivalently be represented as $v = (v_0, \ldots, v_{\lfloor s/2 \rfloor})$ given by $v_i := |\{x, y \in \mathrm{Min}(Q) : x^t Q y = i/s\}|$. No polynomial upper bound in $d$ is known for the scale $s$, however in practice it seems to stay small. The above method is thus efficient for storing this invariant.

An alternative would be to just hash everything. However, the information in the simple numerical invariants such as $|\mathrm{Min}\, Q|$ and $|\mathrm{Aut}(Q)|$ can be very useful. For example to select easy perfect forms to explore or to determine if we can use the basic dual description algorithms (if $|\mathrm{Min}\, Q|$ and $|\mathrm{Aut}(Q)|$ are small) or if we need to exploit the available symmetry (if $|\mathrm{Aut}(Q)|$ is large).

6.3. **Canonical perfect forms.** The second strategy is to construct a canonical form for each perfect form. So we need a function $\varphi$ that maps perfect forms to an arithmetically equivalent perfect form such that if $Q, Q'$ are arithmetically equivalent, then $\varphi(Q) = \varphi(Q')$. One way to achieve this is using Minkowski reduction. However, in practice, obtaining a Minkowski reduced perfect form is too inefficient for our task.

Dutour Sikirić conceived an idea [DS18] for constructing a canonical form based on the canonical graph problem. The problem of finding a canonical representative of a labelled graph has been studied for a long time. Although all practical algorithms are exponential in the worst case, they are very efficient on most

graphs [BA14]. Furthermore, efficient implementations exist in the BLISS [JK07] and Nauty [BA14] software.

Another function we need is a canonical $\mathbb{Z}$-basis extraction function `extract`. Let $V \in \mathbb{Z}^{d\times k}$ be a matrix with $k$ columns that contain a $\mathbb{Z}$-basis of $\mathbb{Z}^d$. We want `extract` to return a $d \times d$ submatrix $W \subset V$ with columns that form a $\mathbb{Z}$-basis of $\mathbb{Z}^d$. Furthermore the function must be canonical, i.e. for any $U \in \mathrm{GL}_d(\mathbb{Z})$ we must have

$$\texttt{extract}(U \cdot V) = U \cdot \texttt{extract}(V).$$

As most properties are preserved by a transformation $U \in \mathrm{GL}_d(\mathbb{Z})$, it is not too hard to construct such a function. In particular we could keep track of a partial basis $B$ and enumerate over the columns of $V$ in a fixed order. Let $v$ be the current enumerated column vector of $V$. If $v$ is linear independent from $B$ we can just add $v$ to $B$. Otherwise we can express $v$ uniquely as $\sum_{b_i \in B} c_i \cdot b$. Now suppose we had started with $U \cdot V$, then we would have a partial basis $U \cdot B$ and we would be considering $Uv$, assuming the previous iterations were canonical. But then the coefficients $c_i$ are the same for all $U \in \mathrm{GL}_d(\mathbb{Z})$. So if we only use these coefficients to make a decision this iteration, and by induction the whole function, automatically becomes canonical. In case all the coefficients are integral we don't need to add $v$ to $B$. Otherwise we use these coefficients to decide which element of $B$ is replaced by $v$.

We start with an overview of the algorithm by Dutour Sikirić [DS18] and then we check why it is canonical. The important property we can use from any perfect form $Q \in \mathcal{S}^d_{>0}$ is that, up to scaling, it is fully determined by its minimal vectors $\mathrm{Min}\, Q$. So if we fix $\lambda(Q) = 1$ it is enough to obtain a canonical set $U \cdot \mathrm{Min}\, Q$ of minimal vectors for some $U \in \mathrm{GL}_d(\mathbb{Z})$. To do this we need to look at a possible larger set $M \supseteq \mathrm{Min}\, Q$ that contains a $\mathbb{Z}$-basis for $\mathbb{Z}^d$. As this isn't always the case for $\mathrm{Min}\, Q$ we can take $C \geq \lambda(Q)$ minimal such that

$$M := \{x \in \mathbb{Z}^d - \{0\} : Q[x] \leq C\}$$

contains a $\mathbb{Z}$-basis of $\mathbb{Z}^d$.

To extract a canonical $\mathbb{Z}$-basis from $M$ we first need to obtain a canonical ordering on $M$. This is were we use a canonical graph algorithm. We construct the graph $G(M)$ where each vertex corresponds to an $x \in M$. Between each two vertices of $G(M)$ corresponding to $x, y \in M$ we have an edge with weight $x^t Q y$. So $G(M)$ is a complete edge-labelled graph. Note that if $Q$ and $Q'$ are arithmetically equivalent, then the corresponding graphs are isomorphic.

We run a canonical graph algorithm on $G(M)$ and get in particular an ordering on $M$. Let $N \in \mathbb{Z}^{d\times k}$ have $k := |M|$ columns with the elements of $M$ in that ordering. Then we extract a $\mathbb{Z}$-basis $W = \texttt{extract}(N)$ of $\mathbb{Z}^d$. Note that $W \in \mathrm{GL}_d(\mathbb{Z})$. We obtain the canonical set $W^{-1} \cdot \mathrm{Min}\, Q$ of minimal vectors that corresponds to the canonical perfect form $W^t Q W$, which we return.

Now we take a look at the correctness of this algorithm. Because $W \in \mathrm{GL}_d(\mathbb{Z})$, the algorithm does correctly return a perfect form arithmetically equivalent to $Q$. Note that any $U \in \mathrm{Aut}(Q)$ induces a permutation on $M = \{x_1, \ldots, x_k\}$ and in particular a graph automorphism on $G(M)$. However, for this graph the correspondence also goes the other way. For this let $\sigma \in \mathrm{Sym}_k$ be any permutation on $M$ that induces a graph automorphism on $G(M)$. So for any $i, j \in [k]$ we have $x_i^t Q x_j = x_{\sigma(i)}^t Q x_{\sigma(j)}$. Now let us assume without loss of generality that the vectors

$x_1, \ldots, x_d$ form a $\mathbb{Z}$-basis of $\mathbb{Z}^d$. Then the permutation $\sigma$ induces a transformation $U \in \mathrm{GL}_d(\mathbb{R})$ such that $Ux_i = x_{\sigma(i)}$ for $i \in [d]$. Because the vectors $x_{\sigma(1)}, \ldots, x_{\sigma(d)}$ are integral, they must generate some sublattice $L = U \cdot \mathbb{Z}^d$ of $\mathbb{Z}^d$ over $\mathbb{Z}$. Let $X = [x_1, \ldots, x_d]$, then $X$ is invertible and we have

$$X^t Q X = X^t U^t Q U X.$$

This implies that $\det(U) = \pm 1$ and thus $L$ is a sublattice of $\mathbb{Z}^d$ of index 1, i.e. $L = \mathbb{Z}^d$. So the vectors $x_{\sigma(1)}, \ldots, x_{\sigma(d)}$ form a $\mathbb{Z}$-basis of $\mathbb{Z}^d$. In particular $U \in \mathrm{GL}_d(\mathbb{Z})$ as $U$ maps a $\mathbb{Z}$-basis of $\mathbb{Z}^d$ to a $\mathbb{Z}$-basis of $\mathbb{Z}^d$. Furthermore for any $j \in [k]$ and all $i \in [d]$ the inner products

$$(Ux_i)^t Q x_{\sigma(j)} = x_i^t Q x_j$$

are fixed, which gives the unique solution $x_{\sigma(j)} = Ux_j$. So $\sigma$ is exactly the graph automorphism on $G(M)$ induced by $U$. As $U$ acts invariantly on $M$ and in particular the set $\operatorname{Min} Q \subset M$, it is an automorphism of $Q$.

Let $Q, Q' \in \mathcal{S}^d_{>0}$ be arithmetically equivalent perfect forms. Let $M$ and $M'$ be the sets from the algorithm for $Q$ and $Q'$ respectively. First note that there exists a transformation $U \in \mathrm{GL}_d(\mathbb{Z})$ such that $Q = U^t Q' U$ and as a result $U \cdot M = M'$, which is fixed up to $\mathrm{Aut}(Q)$. In particular $G(M)$ and $G(M')$ are isomorphic as edge-labelled graphs. The canonicalized versions of the graphs $G(M)$ and $G(M')$ are thus the same up to a graph automorphism. But as these automorphism correspond exactly to the automorphisms $\mathrm{Aut}(Q)$ of $Q$, we can choose $U$ such that if we have the canonical ordering $M = \{x_1, \ldots, x_k\}$ obtained from $G(M)$, then $M' = \{x'_1 = Ux_1, \ldots, x'_k = Ux_k\}$ is the canonical ordering obtained from $G(M')$. Now if $W = \texttt{extract}([x_1, \ldots, x_k])$, then $W' = U \cdot W = \texttt{extract}([x'_1, \ldots, x'_k])$. To conclude

$$W^t Q W = W^t U^t Q' U^t W = (W')^t Q' W'$$

and thus the returned perfect forms are identical.

## 7. Group computations

To make our computations feasible, it is necessary to make use of symmetry. By using face-defining sets, we can assume that we are working with a group $G \subset \mathrm{Sym}_m$ that acts on subsets of $[m]$. So instead of working with all the faces directly, we mostly work with orbit representatives. We need two important tools to make this possible. First, we efficiently want to convert orbit representatives under a group to orbit representatives under another group. Secondly, given a list of orbit representatives we want to determine if an orbit is already represented.

7.1. **Orbit fusing and splitting.** Suppose we have two groups $G_1, G_2 \subset \mathrm{Sym}_m$ and a list $L \subset \mathcal{P}([m])$ of faces on which both groups act invariantly. A common problem is that we only have a non-redundant list $L_1$ of $G_1$-orbit representatives of $L$ and we want to obtain a non-redundant list $L_2$ of $G_2$-orbit representatives of $L$. Often $L$ is too large to enumerate. We consider the two most common cases $G_1 \subset G_2$ and $G_2 \subset G_1$. Note that if we can apply these two conversions, we can apply any conversion in two steps via $G_1 \cap G_2$ or $\langle G_1, G_2 \rangle$. We assume that $L_1 = \{F_1, \ldots, F_k\}$ with representatives $F_i \subset [m]$ of distinct orbits under the action of $G_1$.

First, we consider the case that $G_1 \subset G_2$. Note that any $G_1$-orbit is fully contained in a $G_2$-orbit, in particular we can construct $L_2$ such that $L_2 \subset L_1$. Therefore, to obtain $L_2$ we only need to remove duplicate $G_2$-orbits from $L_1$. We start with $L_2 = \{F_1\}$. Then, for $i = 2, \ldots, k$ we add $F_i$ to $L_2$ if $F_i$ is not $G_2$-equivalent to any representative already in $L_2$. In section 7.2 we consider techniques to check for equivalence.

Secondly, we consider the case that $G_2 \subset G_1$. Then, a priori, it is clear that every $G_1$-orbit $\mathrm{Orb}(G_1, F_i)$ needs to be split into some $G_2$-orbits. To efficiently split orbits without duplicates we can use for every $F_j \in L_1$ the double coset decomposition

$$G_1 = \bigcup_{i=1}^{r} G_2 g_i \mathrm{Stab}(G_1, F_j).$$

Here $g_1, \ldots, g_r$ are elements of $G_1$ and the union is disjoint. There exist algorithms that obtain this decomposition efficiently. One such algorithm is implemented in the GAP algebra software [Gro17]. From the double coset decomposition follows the disjoint orbit decomposition

$$\mathrm{Orb}(G_1, F_j) = \bigcup_{i=1}^{r} \mathrm{Orb}(G_2, g_i F_j)$$

and thus the $G_1$-orbit $\mathrm{Orb}(G_1, F_j)$ splits into $r$ distinct $G_2$-orbits, represented by $g_i F_j$ for $i \in [r]$. We repeat this for every $F_j \in L_1$ to obtain a complete list $L_2$ of $G_2$-orbit representatives of $L$ without duplicates. The double coset decomposition is just the standard right coset decomposition of $G_1$ in $G_2$ if $\mathrm{Stab}(G_1, F_j)$ is trivial. So we only need to calculate that decomposition once. This can speed up the computation significantly if a lot of stabilizers are trivial.

7.2. **Keeping track of orbits.** An important part of working with symmetry is keeping track of found orbits and to check if an orbit has already been found. We consider two natural ways to approach this problem, just as with the perfect forms in Section 6. Let $a \subset [m]$ be a new representative and let $L \subset \mathcal{P}([m])$ be a list of orbit representatives already found.

7.2.1. *Equivalence check.* In general to check if the orbit $\mathrm{Orb}(G,a)$ is already represented in $L$, we need to check if $\mathrm{Orb}(G,a) \cap L$ is empty or not. We can enumerating all $b \in Orb(G,a)$ and check if $b \in L$. However, the size of $\mathrm{Orb}(G,a)$ can be very large, so just enumerating this full set can be infeasible.

The other possibility is to check for every representative $b \in L$ if $b \in \mathrm{Orb}(G,a)$. So, we have to determine if $g \cdot a = b$ for some permutation $g \in G$. There are backtrack algorithms to find such a permutation or prove that such a permutation does not exist, without enumerating the whole group. There are several library implementations, one of them can be found in the GAP algebra software [Gro17].

These backtrack algorithms, although much faster than a full enumeration, are still relatively slow. Therefore, it can be useful to first check some fast $G$-invariants to quickly drop some candidates of $L$. A rather trivial invariant is that the sizes of $a$ and $b$ must coincide. Also we have $|\mathrm{Stab}(G,a)| = |\mathrm{Stab}(G,b)|$. We can save these invariants to quickly obtain a smaller list of candidates. Unfortunately the complexity of this approach still grows significantly as $L$ becomes large. Therefore, this method is not efficient when we have to deal with many orbits.

7.2.2. *Minimal and Canonical image.* The problem of determining if an orbit is already represented in $L$, is basically a lookup on the list $L$. So it is natural to try to get a logarithmic or even amortized constant complexity in the size of $L$ using binary search or hash tables respectively. However, to achieve this it seems we have to reduce our search to a simple check if $a \in L$. Of course, if the orbit $\mathrm{Orb}(G,a)$ is indeed already represented in $L$, this only gives the right answer if that representative is $a$. What we need is a function $\theta : \mathcal{P}([m]) \to \mathcal{P}([m])$ that is canonical in the following way:

(1) $\theta(x) \in \mathrm{Orb}(G,x)$ for all $x \in \mathcal{P}([m])$;

(2) if $x, y \in \mathrm{Orb}(G,x)$ then $\theta(x) = \theta(y)$ for all $x, y \in \mathcal{P}([m])$.

Assuming that $L = \{\theta(b) : b \in L\}$, the check if the orbit $\mathrm{Orb}(G,a)$ is already represented in $L$ boils down to $\theta(a) \in L$. If this is not the case, we add $\theta(a)$ to $L$. Given the right data structure for $L$ lookup and insertion can be done in $O(\log(|L|))$ or even amortized constant time. Therefore, the complexity of this method will mostly be determined by the complexity of computing $\theta(a)$.

One such a canonical function is the minimal orbit representative function. Given a total ordering $\leq$ on $[m]$, we define a total ordering $\preccurlyeq$ on $\mathcal{P}([m])$ by saying that $A \prec B$ if $A$ contains an element $a \notin B$ such that $a \leq b$ for all $b \in B \setminus A$ and $A \preccurlyeq B$ if $A = B$ or $A \prec B$. For simplicity we assume from now on that $\leq$ is the usual ordering. Note that for sets of the same size, the ordering $\preccurlyeq$ is the lexicographic ordering. An important advantage of this ordering compared to the lexicographic ordering is that $A \cap [k] \prec B \cap [k]$ implies that $A \prec B$ for any $k \in [m]$. We define the minimal image function

$$\begin{aligned} \theta_m : \mathcal{P}([m]) &\to \mathcal{P}([m]) \\ a &\mapsto \theta_m(a) := \min_{\preccurlyeq}(\mathrm{Orb}(G,a)). \end{aligned}$$

Computing this function without enumerating the whole orbit, the Minimal image problem, was first treated by Linton [Lin04], who also introduced an algorithm to solve it. We give an overview of how the algorithm works. Let $G = G_0 \supset G_1 \supset \ldots \supset G_m$ be a stabilizer chain of $G$ where $G_i = \mathrm{Stab}(G_{i-1}, i)$ for $i \in [m]$. Note that $G_m$ is the trivial group. Such a chain can be constructed efficiently and generators for the $G_i$ can be saved in the data structure known as

strong generating sets. Strong generating sets in particular make it easy to obtain a representative element of any coset $G_i g$ for $g \in G_{i-1}$. Suppose we have a subset $S \subset [m]$ of which we want to determine its minimal image $M = \theta_m(S)$. There must exist a permutation $g \in G = G_0$ such that $gS = M$. Consider the coset decomposition

$$G_0 = G_1 g_1 \cup \ldots \cup G_1 g_k$$

and note that $g \in G_1 g_i$ for some $i \in [k]$. So the problem of finding the minimal image of $S$ under the action of $G$ is reduced to finding the minimal image of $S$ under the action of each coset $G_1 g_i$. The latter is equivalent to finding the minimal image of $g_i S$ for all $i \in [k]$ under the action of $G_1$. So we obtain $k$ candidates $g_1 S, \ldots, g_k S$ of which at least one has minimal image $M$ under $G_1$. Note that $G_1$ keeps 1 invariant, so if $1 \notin g_i S$, then 1 cannot be in any element of the orbit $\mathrm{Orb}(G_1, g_i S)$. Therefore, if at least one of the candidates $g_i S$ contains 1, we can remove all candidates $g_i S$ that do not contain 1, as they will necessarily have a larger minimal image under $G_1$. We repeat this method with the cosets of $G_i$ in $G_{i-1}$ until $G_i$ is trivial. Instead of starting with the single candidate $S$, we start with all the candidates that were not removed in the previous iteration. Whenever $G_i$ is trivial, $M$ must be equal to the only remaining candidate.

When $G_i$ already stabilizes the elements $i+1, \ldots, i+k$, we can immediately skip to the cosets of $G_{i+k}$ in $G_i$. The main problem is that the number of candidates can become very large in some iterations. In a recent publication by Jefferson, Jonauskyte, Pfeiffer and Waldecker [JJPW18] this inefficiency is tried to be solved. We give an overview of their accomplishments.

Improvements were introduced to cheaply reject more candidates in each iteration, by using properties of the orbit decomposition of $[m]$ under the action of $G_i$. It is discussed that the number of candidates in intermediate steps is highly dependent on $S$ and the ordering used on $[m]$. An algorithm that finds a good ordering given a group $G$ is introduced. However, it is also shown that any fixed ordering remains to be problematic for some inputs. Therefore, a new approach for a canonical function is considered that uses a dynamical ordering. Each orbit can potentially have a different ordering and the algorithm returns the minimal orbit representative under that ordering. Instead of a fixed ordering up front, the dynamical ordering is constructed incrementally in each iteration of the algorithm. This ordering is tailored for the orbit in question in a canonical way, i.e. the constructed ordering is the same for any pair of representatives of the same orbit. The ordering is constructed with as goal to minimize the number of candidates during the algorithm.

Multiple heuristics are considered to incrementally create a canonical ordering and it is shown experimentally which heuristics perform the best. From now on we assume that we have chosen one of those well performing heuristics which fixes our canonical function $\theta_c$. In practice, computing $\theta_c$ performs up to a few orders of magnitude faster than $\theta_m$ for large groups. For experimental comparisons and more details see the publication by Jefferenson, et al. [JJPW18]. An implementation by the authors is available in the GAP package Images [JJPW].

## 8. Conversion of polyhedral cone representations

By the representation theorem of cones, a cone $\mathcal{C}$ can either be written as an intersection of half-spaces $\mathcal{C} = P(A, 0)$ or as generated by some (extreme) rays $\mathcal{C} = \text{cone}(Y)$. Given a cone $\mathcal{C} = P(A, 0)$ a natural problem is to compute the set $Y$ of extreme rays of $\mathcal{C}$. By duality the other direction is equivalent. This problem has several names such as the dual description problem or the representation conversion problem. There are two main classes of algorithms to solve the representation conversion problem: the double description method and the reverse search algorithm. The double description method starts with a simple cone and adds the inequalities that define $\mathcal{C}$ one by one, while keeping track of the extreme rays. The reverse search algorithm moves from extreme ray to extreme ray, while using a 'reverse pivoting' trick to find every extreme ray exactly once.

### 8.1. Double Description method.

The Double Description method is heavily inspired by a proof of the representation theorem using so-called Fourier-Motzkin elimination [Mot53]. Because the Double Description method is very natural, it was rediscovered multiple times and known under multiple names such as the Double Description method [Mot53, FP96] and Chernikova's algorithm [Che65, LV92].

We give a short outline of the algorithm as shown by Le Verge [LV92]. We consider the cone $\mathcal{C} = P(A, 0) \subset \mathbb{R}^n$ with the inequality matrix $A$ having row vectors $a_1, \ldots, a_m \in \mathbb{R}^n$. For simplicity we assume that $\mathcal{C} = P(A, 0) \subset \mathbb{R}^n_{\geq 0}$. Let $H_1, \ldots, H_m$ be half-spaces where $H_k$ is defined by $\{x \in \mathbb{R}^n : a_k \cdot x \geq 0\}$. The Double Description method works iteratively by constructing the following cones

$$\begin{aligned} \mathcal{C}_0 &= \mathbb{R}^n_{\geq 0} \\ \mathcal{C}_k &= \mathcal{C}_{k-1} \cap H_k \qquad\qquad \text{for } k \in [m], \end{aligned}$$

while keeping track of the set $E_k$ of extreme rays of $\mathcal{C}_k$. Finally $P = \mathcal{C}_m$ and thus $E_m$ gives the set of extreme rays of $P$. The set $E_k$ of extreme rays of $\mathcal{C}_k$ is constructed from the set $E_{k-1}$ of extreme rays of $\mathcal{C}_{k-1}$, starting from the set $E_0 = \{e_1, \ldots, e_n\}$ of extreme rays of $\mathbb{R}^n_{\geq 0}$. If we intersect $\mathcal{C}_{k-1}$ with a half-space $H_k$, any extreme ray of $\mathcal{C}_{k-1}$ that lies fully inside $H_k$ is again an extreme ray of $\mathcal{C}_k$. We call two extreme rays adjacent if the 2-dimensional cone they span is a face of $\mathcal{C}_{k-1}$. Any pair of adjacent rays of $\mathcal{C}_{k-1}$ of which one lies strictly inside $H_k$ and the other lies strictly outside $H_k$ also give a new extreme ray of $\mathcal{C}_k$. The new extreme ray of $\mathcal{C}_k$ is exactly the intersection of the 2-dimensional face spanned by the adjacent rays and the hyperplane corresponding to $H_k$. To summarize we define the following sets

$$\begin{aligned} E^{=}_{k-1} &= \{y : y \in E_{k-1}, a_k \cdot y = 0\}, && \text{(Boundary)} \\ E^{>}_{k-1} &= \{y : y \in E_{k-1}, a_k \cdot y > 0\}, && \text{(Inside)} \\ E^{<}_{k-1} &= \{y : y \in E_{k-1}, a_k \cdot y < 0\}. && \text{(Outside)} \end{aligned}$$

Then the set $E_k$ is given by

$$E_k = E^{=}_{k-1} \cup E^{>}_{k-1} \cup \overline{E_{k-1}} \qquad\qquad \forall k \in [m]$$

where $\overline{E_{k-1}}$ is defined by

$$\{y : a_k \cdot y = 0, y = \lambda y_1 + \mu y_2, (y_1, y_2) \in E^{>}_{k-1} \times E^{<}_{k-1}, y_1, y_2 \text{ adjacent}, \lambda > 0\}.$$

If we remove the adjacency condition from $\overline{E_{k-1}}$ we do obtain rays of $\mathcal{C}_k$, but these rays aren't necessarily extremal. The major differences between implementations

and their efficiency is determined by the construction of $\overline{E_{k-1}}$ [LV92, FP96]. There are several libraries implementing the Double Description method such as CDD [Fuk93] and the Parma Polyhedra Library [BHZ08].

8.2. **Reverse search method.** For simplicity we explain the reverse search method in the setting of enumerating vertices of a polytope. The reverse search method was conceived by Avis and Fukuda [AF92]. It makes use of the fact that, if we optimize a linear function over a polytope using a simplex method, we implicitly create a path from each vertex to an optimal vertex. We call a set of $n$-independent facet defining inequalities a basis of a vertex if they all give an equality at this vertex. We first consider the case of a simple polytope, a polytope for which each vertex lies in exactly $n$ facets. Then each vertex has a unique basis.

Suppose we fix a pivoting technique, a rule that determines to which neighbouring vertex we move to improve the function value. More detailed, the pivoting rule determines which basis element has to be removed and added to improve the function value. Also assume our function is optimal at a unique vertex. Then we get from each vertex a unique path to the optimal vertex by following the pivot rule. Reversing these paths gives a spanning tree of all vertices with the optimal vertex at its root. See Figure 10 for an example by Avis [Avi00]. The reverse search method is a backtrack algorithm that enumerates this tree.

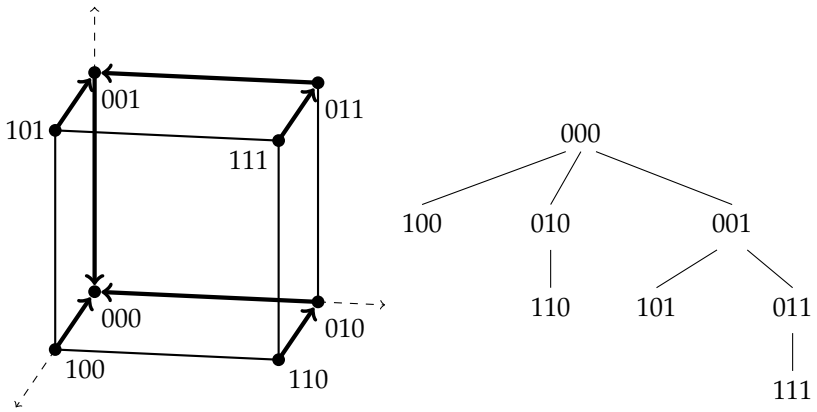


FIGURE 10. Example by Avis [Avi00] of a reverse search tree construction with objective function $-\sum_{i=1}^{3} x_i$. The arrows show the pivot direction.

To enumerate this tree, we must to be able to move up and down in the tree. Moving up is easy, as we just apply the pivot rule to get closer to the root. Moving down, the so-called reverse search step, is more expensive. From our current vertex, we need to find neighbouring vertices, such that pivoting brings us back at our current vertex. We remove a basis element which gives us an edge of the polytope to another vertex. We determine this neighbouring vertex and then apply the pivot rule to check if it brings us back. If pivoting brings us back we move down to this neighbouring vertex in the tree.

It is possible to implement this strategy, while only keeping track of the current vertex and its basis. Because, if we move up, we immediately know which basis element we treated last in our reverse search. Namely, the basis element that is removed by the pivoting. Then we can continue by removing the next basis element. Therefore, the memory usage of the reverse search method does not depend on the number of vertices, which can become really large.

When the polytope is not simple, each vertex can have a lot of different bases and the above algorithm will find all these bases. To prevent this, Avis [Avioo] improved the previous algorithm by defining a unique so-called minimal basis for each vertex. Furthermore he used a pivoting rule that, given a minimal basis, always pivots to a minimal basis and in the reverse search an extra condition is added that the new basis must be minimal. In this way only the minimal bases are enumerated instead of all bases. In particular we have to execute the expensive reverse search operation less often. The reverse search method is implemented by Avis in the software LRS [Avioo].

8.3. **Symmetries in Voronoi's algorithm.** Unfortunately for highly degenerate $n$-dimensional cones given by $m > n$ inequalities, the number of extreme rays can be extremely high. This makes any method that produces a list of all these rays infeasible. Fortunately, the highly degenerate cones $\mathcal{P}(Q)$ that we encountered in Voronoi's algorithm so far, also have a large automorphism group induced by $\mathrm{Aut}(Q) \subset \mathrm{GL}_d(\mathbb{Z})$. For any extreme ray $R$ of $\mathcal{P}(Q)$ and automorphism $U \in \mathrm{Aut}(Q)$

$$Q + \alpha U^t R U = U^t (Q + \alpha R) U$$

is arithmetically equivalent to $Q + \alpha R$. So $R$ and $U^t R U$ lead to the same contiguous perfect form up to arithmetical equivalence. Therefore, it is enough to only determine the orbits of the extreme rays of the cone $\mathcal{P}(Q)$ under the action of $\mathrm{Aut}(Q)$.

For any automorphism $U \in \mathrm{Aut}(Q)$, we have $U \cdot \mathrm{Min}\, Q = \mathrm{Min}\, Q$, so the automorphism group $\mathrm{Aut}(Q)$ induces a permutation group on $\mathrm{Min}\, Q$. In particular if we fix an ordering on the minimal vectors $\mathrm{Min}\, Q = \{\pm x_1, \ldots, \pm x_m\}$, then $\mathrm{Aut}(Q)$ induces a permutation group $\mathrm{AutMin}(Q) \subset \mathrm{Sym}_m$ on $[m]$. Because of the automorphism $-I_d \in \mathrm{Aut}(Q)$, which induces the trivial permutation, $\mathrm{AutMin}(Q)$ has half the order of $\mathrm{Aut}(Q)$. This embedding depends on the order of the minimal vectors, which we make clear in the context, whenever it matters. As the action of $U \in \mathrm{Aut}(Q)$ on $\mathcal{S}^d$ is linear on the vector of all matrix coefficients we have

$$\mathrm{AutMin}(Q) \subset \mathrm{Lin}_A(\mathcal{P}(Q)) \subset \mathrm{Comb}(\mathcal{P}(Q))$$

with $A = (x_i x_i^t)_{1,\ldots,m}$. By these inclusions it is well defined to obtain orbits of the extreme rays of $\mathcal{P}(Q)$ under the action of $\mathrm{AutMin}(Q)$.

Because of this symmetry, Voronoi's algorithm could be completed for dimensions 6, 7 and 8 [Bar57, JC93, DSSV07]. In particular it was necessary to determine the neighbours of the optimal perfect forms corresponding to the root lattices $E_6, E_7$ and $E_8$ [Bli35]. The significant automorphism group of the perfect form $Q_{E_8}$, reduced the total number of 25.075.566.937.584 extreme rays of the cone $\mathcal{P}(Q_{E_8})$, to only 83.092 orbits under the action of $\mathrm{AutMin}(Q_{E_8})$ [DSSV07]. For dimensions 7 and 8 this symmetry was exploited using the Adjacency Decomposition Method, which is covered in Section 8.4. If needed, we can also use the possibly larger automorphism group $\mathrm{Lin}_A(\mathcal{P}(Q))$. Then we use the orbit splitting methods from section 7.1 to obtain the orbits of extreme rays under the action of $\mathrm{AutMin}(Q)$.

8.4. **Adjacency Decomposition Method.** The Adjacency Decomposition Method can be seen as an adaptation of Voronoi's algorithm to a cone given by a finite number of inequalities or extreme rays. It was rediscovered multiple times, for example by Jaquet-Chiffelle [JC93], Christof and Reinelt [CR96] and Deza, Fukuda,

Pasechnik and Sato [DFPSoo]. The easiest, and by duality equivalent, way to describe the recursive behaviour of the Adjacency Decomposition Method is when the cone is given by its extreme rays and we want to obtain its facets.

Let $\mathcal{C} = \text{cone}(Y) \subset \mathbb{R}^n$ be a full dimensional cone generated by $m$ extreme rays $Y = [y_1, \ldots, y_m] \in \mathbb{R}^{n \times m}$. Remind yourself that in this subsection a face-defining set of a face $F \subset \mathcal{C}$ is the set of all extreme rays contained in $F$ and $\text{Comb}(\mathcal{C})$ is the maximal permutation group acting invariantly on the set of face-defining sets. of $\mathcal{C}$.

**Input:** A cone $\mathcal{C}$ given by $m$ extreme rays and a group $G \subset \text{Comb}(\mathcal{C}) \subset \text{Sym}_m$.
**Output:** A complete list $\mathcal{F}$ of $G$-inequivalent facets of $\mathcal{C}$.

$\mathcal{T} \leftarrow \{F\}$ with $F$ a facet of $\mathcal{C}$;
$\mathcal{F} \leftarrow \emptyset$;
**while** *there exists a facet* $F \in \mathcal{T}$ **do**
  $\mathcal{F} \leftarrow \mathcal{F} \cup \{F\}$;
  $\mathcal{T} \leftarrow \mathcal{T} \setminus \{F\}$;
  $\mathcal{H} \leftarrow$ ridges of $\mathcal{C}$ in $F$;
  **for** $H \in \mathcal{H}$ **do**
    Obtain $F'$ such that $H = F \cap F'$;
    **if** $F'$ *is $G$-inequivalent to all facets in* $\mathcal{F} \cup \mathcal{T}$ **then**
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{F'\}$;

**Algorithm 4:** Adjacency Decomposition Method.

The Adjacency Decomposition Method shown in Algorithm 4 first finds an initial facet. An initial facet can be found by solving a linear program or by a depth first search in the face lattice, using the algorithm shown in section 9.1. Suppose we have an initial facet-defining set $F_1 \subset [m]$ of the cone $C$. The next step is to find all ridges $H_1, \ldots, H_k$ of $\mathcal{C}$ that lie in the facet $F_1$. Given such a ridge $H_i$, it is computationally easy to find the adjacent facet $F_2$ such that $H_i = F_1 \cap F_2$. In this way we find all $k$ adjacent facets of the facet $F_1$. We check if the orbit of a found facet $F_2$ is already represented under $G$ and we discard the facet if this is the case. We repeat what we did for the initial facet, for every new facet that we find, that is not immediately discarded. Because all facets are connected by other facets, the Adjacency Decomposition Method finds a complete list of orbit representatives for the facets of the cone $\mathcal{C}$.

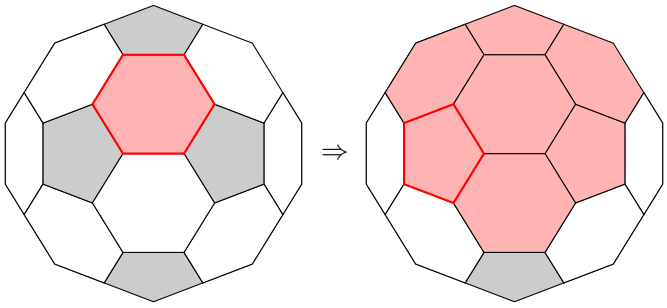

FIGURE 11. Exploration of facets by the Adjacency Decomposition Method in the setting of a polytope.

We shortly explain the so-called gift-wrapping-step to compute the adjacent facet $F_2$, given the facet $F_1$ and a ridge $H \subset F_1 \subset [m]$. We follow the explanation by Schürmann [Scho9]. Any defining inequality $a \in \mathbb{R}^m$ of the facet $F_2$ should satisfy $a^t y_i = 0$ for all extreme rays $i \in H$. Let $\{a_1, a_2\}$ be a basis of the 2-dimensional vector space of such inequalities. Suppose that $a = \alpha_1 a_1 + \alpha_2 a_2$ is the inequality defining $F_1$ or $F_2$, then $a^t y_i \geq 0$ for all $i \in [m]$. But this gives $m$ mostly redundant linear inequalities on $\alpha_1, \alpha_2$, which define a 2-dimensional cone. The two extreme rays of this cone, which are easily obtained as we are working in 2-dimensional space, give us exactly the facet-defining inequalities of $F_1$ and $F_2$.

Every facet $F$ of the cone $\mathcal{C}$ we find is itself a cone $\mathcal{C}' \subset \mathcal{C}$ of dimension $n-1$. So finding the ridges of the facet $F$ is equivalent to finding the facets of the cone $\mathcal{C}'$. Therefore, it is again a dual description problem. However the dimension decreased by 1 and more importantly the number of extreme rays defining the cone $\mathcal{C}'$ is exactly the number of extreme rays of $\mathcal{C}$ contained in $\mathcal{C}'$. We only have the guarantee that there are at most $m-1$ of such extreme rays, but in practice the number is often much lower. Then the cone $\mathcal{C}'$ is less degenerate than the cone $\mathcal{C}$, so we can either determine the facets of the cone $\mathcal{C}'$ with the Double Description method or the reverse search method. If the cone $\mathcal{C}'$ is still too degenerate, we can recursively apply the Adjacency Decomposition Method using the group $\mathrm{Stab}(G, F) \subset \mathrm{Comb}(C')$. The full combinatorial automorphism group $\mathrm{Comb}(C')$ of the cone $\mathcal{C}'$ can be much larger than $\mathrm{Stab}(G, F)$. So it could be useful to first compute a hopefully larger linear automorphism group of the cone $\mathcal{C}'$ and again use the orbit splitting methods from Section 7.1 to obtain the facets of the cone $\mathcal{C}'$ under the action of $\mathrm{Stab}(G, F)$.

8.5. **Implementation details.** The Adjacency Decomposition Method is implemented in the GAP package Polyhedral by Dutour Sikirić [DS13]. An early version of this package was used to obtain the dual description of the cone $\mathcal{P}(Q_{E_8})$ under the action of $\mathrm{AutMin}(Q_{E_8})$. As a result Voronoi's algorithm in dimension 8 could finally be concluded [DSSV07].

If we have the inclusions $F' \subset F_1 \subset \mathcal{C}$ for a ridge $F'$ and a facet $F_1$, then there exists an adjacent facet $F_2$ such that $F' \subset F_2 \subset \mathcal{C}$. Suppose we apply the Adjacency Decomposition Method to the cone $\mathcal{C}$ and it has to recurse on both facets $F_1$ and $F_2$. Then the dual description of the ridge $F'$ is computed twice. As the recursion depth grows, the number of such repetitions increases. Therefore, the implementation in the Polyhedral software already has a banking system to save such dual description computations. It is easy to change the heuristics in the Polyhedral software about which computation to save and when to recursively use the Adjacency Decomposition Method.

We made some improvements to the Polyhedral package to make the computations more efficient. For example this reduced the time to compute the extreme rays of $\mathcal{P}(Q_{E_8})$ to under two weeks of computations on a single core, instead of the 15 months it originally took [DSSV07]. Although the latter is, of course, on a slower core, we certainly saw a big speed-up in our own computations after these improvements.

First, for obtaining the dual description of base cases we used the Double Description method implementation of the Parma Polyhedral Library [BHZ08], instead of the much older and slower CDD [Fuk93]. As a result much more degenerate cases could efficiently be solved without the Adjacency Decomposition Method.

We also altered the recursion heuristics of the Polyhedral package to take this into account.

Secondly, a major part of the computations in the Adjacency Decomposition Method involve determining if an orbit under a permutation group $G$ has already been found. For large groups the canonical minimal image function $\theta_m$ is too slow. So the Polyhedral package uses the first method explained in Section 7.2.1 to deal with orbits of large groups. However, this method does not scale if the number of orbits grows. The recently conceived canonical image function $\theta_c$, mentioned in Section 7.2.2, solves this problem by being an efficient alternative to the minimal image function. We adapted the Polyhedral package to keep track of orbits using the canonical image function.

## 9. Face enumeration under symmetry

We adapt the geometrical algorithm by Fukuda, Liebling and Margot [FLM97] for finding all faces of a cone given by its $\mathcal{H}$-representation. Again by switching to the dual, this is equivalent to when a cone is given by its $\mathcal{V}$-representation. Assume that the cone is given by $\mathcal{C} = P(A, 0) = \{x \in \mathbb{R}^n : Ax \geq 0\}$ with the inequality matrix $A$ having row vectors $a_1, \ldots, a_m \in \mathbb{R}^n$. The goal is to only find one representative of every orbit of the faces $L$ under the action of a subgroup of the combinatorial automorphism group $\mathrm{Comb}(\mathcal{C})$. This algorithm does not have a direct usage in Voronoi's algorithm, but it can be used as a tool to explore the structure and possible extra symmetries of the highly degenerate but symmetric cones we encounter.

9.1. **Geometric face enumeration.** We approach the geometric face enumeration problem with a backtrack algorithm described by Fukuda, et al. [FLM97]. In this subsection we give an overview of the algorithm, which we adapt in Sections 9.2 and 9.3 to an algorithm that exploits the available symmetry.

Every face-defining set is only visited once, by using a partitioning strategy. First we enumerate all face-defining sets that contain the first facet, then we enumerate all face-defining sets that contain the second facet, but not the first, and so on. Of course, we apply such a partitioning on every level of the face lattice and obtain therefore a backtrack algorithm that enumerates every face-defining set exactly once.

To apply this strategy we must solve a smaller problem called the Restricted Face of a Polyhedron (RFP). Let $R, S \subset [m]$ be two subsets. The question is: does there exist a face-defining set $F \subset [m]$ for the cone $\mathcal{C}$ such that

$$R \subset F \qquad \text{and} \qquad F \cap S = \emptyset?$$

Equivalently, does there exists a point $x \in \mathcal{C}$ in the cone such that $a_r \cdot x = 0$ for all facets $r \in R$ and $a_s \cdot x > 0$ for all facets $s \in S$. This problem is easily translated to a linear program

| | | |
|---|---|---|
| **Maximize:** | $y$ | |
| **Subject to:** | $a_r \cdot x = 0$ | for all $r \in R$, |
| | $a_s \cdot x \geq y$ | for all $s \in S$, |
| | $a_t \cdot x \geq 0$ | for all $t \in [m] \setminus (R \cup S)$, |
| | $y \leq 1$. | |

Then $\mathrm{RFP}(R, S)$ is true if and only if for the optimal value we have $y > 0$.

If $\mathrm{RFP}(R, S)$ is true, there exists a face-defining set $F$ of the cone $\mathcal{C}$ such that $R \subset F$ and $F \cap S = \emptyset$. In order to not skip any faces in our enumerations, we must find the minimal, under inclusion, face-defining set $F$ that satisfies the above conditions.

The intersection of all facets in $R$ already defines a face $V$ of the cone $\mathcal{C}$. So we want to determine the face-defining set of $V$. Then we need to determine all facets in which $V$ is fully contained, i.e. all facets $i \in [m]$ such that $a_i \cdot x = 0$ for all $x \in V$. As $V$ is defined as the intersection of all facets in $R$ this is exactly coincides with $\mathrm{RFP}(R, \{i\})$ being false. Furthermore, because $\mathrm{RFP}(R, S)$ is true, we certainly have that $\mathrm{RFP}(R, \{i\})$ is true for all facets $i \in S$. As a result the

minimal face-defining set $F$ of the cone $\mathcal{C}$ for which $R \subset F$ and $F \cap S = \emptyset$ is given by

$$\begin{aligned} F = \mathrm{MFDS}(R) :&= \{j \in [m] : \mathrm{RFP}(R, \{j\}) = \mathrm{False}\} \\ &= R \cup \{j \in [m] \setminus (R \cup S) : \mathrm{RFP}(R, \{j\}) = \mathrm{False}\}. \end{aligned}$$

**Input:** $\mathcal{C} = P(A, 0)$ and $R, S \subset [m]$
**Output:** All face-defining sets $F$ of $\mathcal{C}$ such that $R \subset F$ and $F \cap S = \emptyset$

```
1 if RFP(R, S) = true then
2 |   F ← MFDS(R);
3 |   output F;
4 |   J ← [m] − (F ∪ S);
5 |   Let J be ordered as {j_1, j_2, ..., j_t} such that j_1 < ... < j_t;
6 |   for k ← 1 to t do
7 |   |   FaceEnum(F ∪ {j_k}, S ∪ {j_1, ..., j_{k−1}});
```

**Algorithm 5:** FaceEnum$(R, S)$ by Fukuda, et al [FLM97].

We obtain Algorithm 5 that enumerates all face-defining sets of the cone $\mathcal{C}$ exactly once. For correctness let $F$ be face-defining and consider the set of face-defining sets $F'$ such that $F \subsetneq F'$ and $F' \cap S = \emptyset$. This set can be partitioned in the sets $F^{j_1}, \ldots, F^{j_t}$, where $F^{j_k}$ contains all face-defining sets $F'$ such that $F \cup \{j_k\} \subset F'$ and $F' \cap (S \cup \{j_1, \ldots, j_{k-1}\}) = \emptyset$. If the union $F \cup S$ is large enough, there is no face-defining set $F'$ such that $F \subset F'$ and $F' \cap S = \emptyset$. So inductively Algorithm 5 works correctly and enumerates every face-defining set in the scope exactly once.

The complexity of this algorithm is $O(f \cdot m \cdot l(m, n))$, where $f$ is the number of faces in the scope and $l(m, n)$ is the complexity of solving a linear program in dimension $n$ with $m$ (in)equalities.

We define the enumeration tree $T(R_{init}, S_{init})$ for execution of Algorithm 5 with starting values $R_{init}$ and $S_{init}$. We denote each successful FaceEnum call, i.e. such that $\mathrm{RFP}(R, S)$ is true, by a node $(F, S)$ where $F = \mathrm{MFDS}(R)$. The edges correspond to the recursive calls. If the initial values do not matter in the context, we also denote an enumeration tree just by $T$. The root is given by $(\mathrm{MFDS}(R_{init}), S_{init})$, assuming $\mathrm{RFP}(R_{init}, S_{init})$ is true. See Figure 12 for an example of such an enumeration tree.

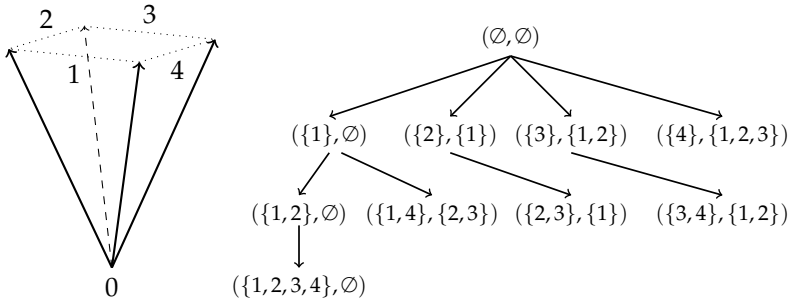


FIGURE 12. Enumeration tree $T(\emptyset, \emptyset)$ of a simple cone given by 4 inequalities.

If the node $(F', S') \in T$ is an ancestor of the node $(F, S) \in T$, using tree terminology, then we have the inclusions $F' \subset F$ and $S' \subset S$. Sometimes, if the exclusion set $S$ does not matter, we also indicate a node by only the face-defining set $F \in T$.

Furthermore, given two nodes $F, F' \in T$, there always exists a maximal, in size of $F$, common ancestor $H \in T$ such that $H \subset F, F'$. In tree terminology this would be the Lowest Common Ancestor. It is tempting to hope that $H = F \cap F' \in T$ is the maximal common ancestor of the face-defining sets $F$ and $F'$, however this is not necessarily the case. For example in Figure 12 the maximal common ancestor of $\{1,2,3,4\}$ and $\{1,4\}$ is $(\{1\},\emptyset)$. We do always have $H \subset F \cap F'$.

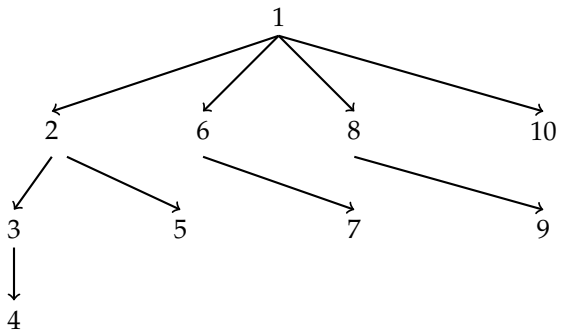


FIGURE 13. Preordering induced by Algorithm 5 on the example in Figure 12.

We say that the node $(F, S) \in T$ is enumerated earlier than the node $(F', S') \in T$, if the node $(F, S)$ comes earlier than the node $(F', S')$, in the depth first preordering induced by Algorithm 5. In particular the root is enumerated first. See Figure 13 for the ordering of the example in Figure 12.

9.2. **Geometric face enumeration using symmetry.** We want to make use of the known symmetry. In particular, we want to determine a single representative for each orbit of the faces of a cone $\mathcal{C}$ under the action of a subgroup $G \subset \mathrm{Comb}(\mathcal{C})$ of the combinatorial automorphism group. We can adapt Algorithm 5 to only enumerate orbits. We keep a list of all found representatives of orbits. Then, for each found face-defining set $F$ in the node $(F, S)$, we check, if the corresponding orbit has already been found. If not we add it and continue our backtrack search.

In case we already found a face $F'$ in the node $(F', S')$ with $F = gF'$ for some permutation $g \in G$, we would like to immediately return. So we do not have to recurse further. If we also have $S = gS'$, it is not hard to see that we do not skip any orbits by returning immediately. However, there is no reason that $S$ does equal $gS'$ and the correctness of the algorithm becomes less clear.

Fortunately, we can even prove something stronger. Not only all orbits are found if we return immediately when the orbit has already been found, but the representative found for each orbit is also lexicographic minimal. Algorithm 5 enumerates all fixed size face-defining sets in this lexicographic ordering. If two face-defining sets $F$ and $F'$ lie in the same orbit, then certainly $|F| = |F'|$. In particular, if we find a face-defining set $F$, the corresponding orbit has not been found before if and only if $F$ is minimal in its orbit.

Remind that $\theta_m(F)$ returns the minimal element in the orbit $\mathrm{Orb}(G, F)$ under the ordering $\prec$, defined in Section 7.2.2. This ordering restricted to sets of the same size is equivalent to the lexicographic ordering, so $F$ is minimal in its orbit if and only if $\theta_m(F) = F$.

**Lemma 12.** *Let $T$ be the enumeration tree of an execution of Algorithm 5 and let $F, F' \in T$ be distinct face-defining sets with $|F| = |F'|$. Then $F \prec F'$ if and only if $F$ is enumerated earlier than $F'$.*

*Proof.* Consider the maximal common ancestor $(F_m, S_m) \in T$ of the face-defining sets $F$ and $F'$. Because $|F| = |F'|$ and $F \neq F'$, we get that $F_m \notin \{F, F'\}$. Let $J = \{j_1, \ldots, j_t\}$ with $j_1 < \ldots < j_t$ be as in Algorithm 5 in node $(F_m, S_m)$. So we have $F \in F_m^{j_{k_1}}$ and $F' \in F_m^{j_{k_2}}$ for some distinct partitions $k_1, k_2 \in [t]$. To conclude, note that $F \prec F'$ if and only if $k_1 < k_2$ if and only if $F$ is enumerated earlier than $F'$. The second equivalence follows trivially from the recursion order of Algorithm 5. To clarify the first equivalence, note that if $k_1 < k_2$, then

$$F_m \subset F \cap F',$$
$$j_1, \ldots, j_{k_1 - 1} \notin F \cup F'.$$

So all facets $i \in [j_{k_1} - 1]$ are either in both or neither one of the face-defining sets $F$ and $F'$. Because $j_{k_1} \in F$ and $j_{k_1} \notin F'$, we get $F \prec F'$.

□

**Input:** $\mathcal{C} = P(A, 0)$ and $G \subset \mathrm{Comb}(P)$
**Output:** Minimal representatives for all face-defining set orbits of $\mathcal{C}$ under the action of $G$.

**Function** *FaceEnumGroupRecursive(F,S)*:
  **if** $RFP(R, S) = true$ **then**
    $F \leftarrow \mathrm{MFDS}(R)$;
    **if** $\theta_m(F) \neq F$ **then**
      **return**;
    output $F$;
    $J \leftarrow [m] - (F \cup S)$;
    Let $J$ be ordered as $\{j_1, j_2, \ldots, j_t\}$ such that $j_1 < j_2 < \ldots < j_t$;
    **for** $k \leftarrow 1$ **to** $t$ **do**
      FaceEnumGroupRecursive$(F \cup \{j_k\}, S \cup \{j_1, \ldots, j_{k-1}\})$;
**Function** *FaceEnumGroup()*:
  FaceEnumGroupRecursive$(\emptyset, \emptyset)$

**Algorithm 6:** FaceEnumGroup().

In particular, if we run FaceEnum$(\emptyset, \emptyset)$, the first representative of each orbit we find is minimal. Suppose that the recursive enumeration stops when the found face if not minimal in its orbit. Then we obtain Algorithm 6. This adapted algorithm can only fail to find an orbit if a minimal orbit representative $F \in T(\emptyset, \emptyset)$ has an ancestor that is not minimal in its orbit. Because that can be the only reason that node $F$ is not reached in Algorithm 6. To show that this is not possible we need Lemma 13.

**Lemma 13.** *Suppose we have face-defining sets $F, F' \in T$ with $F' \subset F$. Then either $F'$ is an ancestor of $F$ or $F$ is enumerated earlier than $F'$.*

*Proof.* Let $(F_m, S_m)$ be the maximal common ancestor of the face-defining sets $F'$ and $F$ in $T$. If $F_m = F'$, then $F'$ is an ancestor of $F$. So suppose that $F_m \neq F'$. As $F$ cannot be an ancestor of $F'$, because $F \not\subset F'$, we have $F_m \notin \{F, F'\}$. Let $J = \{j_1, \ldots, j_t\}$ with $j_1 < \ldots < j_t$ be as in Algorithm 5 in node $(F_m, S_m)$. So we

have $F \in F_m^{j_{k_1}}$ and $F' \in F_m^{j_{k_2}}$ for some distinct partitions $k_1, k_2 \in [t]$. If $k_2 < k_1$, then $j_{k_2} \in F'$ and $j_{k_2} \notin F$, which gives a contradiction, because $F' \subset F$. So $k_1 < k_2$ and thus $F$ is enumerated earlier than $F'$. □

**Theorem 14.** *If the face-defining set $F \in T(\emptyset, \emptyset)$ is minimal in its orbit, then all ancestors $F' \in T(\emptyset, \emptyset)$ of $F$ are minimal in their orbit.*

*Proof.* Note that $F' \subset F$ and that the ancestor $F'$ of $F$ is enumerated before $F$. Assume for contradiction that the face-defining set $F'$ is not minimal in its orbit. Then there exists a permutation $g \in G$ such that $gF' \prec F'$. We show that then $gF \prec F$, which contradicts the minimality of $F$.

Observe that $gF'$ and $F'$ are the roots of two non-overlapping subtrees $T_1$ and $T_2$ of $T(\emptyset, \emptyset)$. Also, $F$ lies in $T_2$. Because $gF'$ is enumerated before $F'$ by Lemma 12, all elements in $T_1$ are enumerated earlier than those in $T_2$. Note that $gF' \subset gF$, so by Lemma 13 $gF'$ is either an ancestor of $gF$, or $gF$ is enumerated earlier than $gF'$. If $gF'$ is an ancestor of $gF$, then $gF$ lies in $T_1$ and thus $gF$ is enumerated earlier than $F$. If $gF$ is enumerated earlier than $gF'$, then we have the enumeration order

$$gF \to gF' \to F' \to F$$

and thus $gF$ is again enumerated before $F$. In both cases we get the contradiction $gF \prec F$ by Lemma 12. □

So Theorem 14 shows that Algorithm 6 finds the minimal representative of each face-defining set orbit of the cone $\mathcal{C}$ under the action of $G$. See Figure 14 to see Algorithm 6 applied to the example in Figure 12. An important property of Algorithm 6 is, that only a constant amount of memory is needed during the enumeration. Because we do not need to save a representative of each found orbit. Furthermore, after some initialization work, it is easy to parallelise without any inter-node communication. A disadvantage of Algorithm 6 is that we need to compute the minimal image function $\theta_m$ often. This can be very expensive, certainly if indeed $\theta_m(F) = F$. If $\theta_m(F) \neq F$, the algorithm that computes $\theta_m(F)$, described in Section 7.2.2, can reject as soon as a smaller candidate than $F$ is found. This is often much faster.

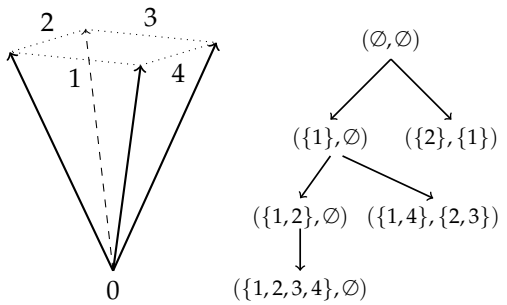


FIGURE 14. Enumeration by Algorithm 6 of minimal orbit representatives of faces under the action of $G = \langle (13)(24) \rangle$.

A way to remove the slow minimal image function is to keep a list $L$ of canonical representatives of the already found orbits, using the canonical image function $\theta_c$. Then instead of checking if $\theta_m(F) \neq F$, we check if $\theta_c(F) \in L$. Using the canonical image function, instead of the minimal image function, is for large groups much more efficient, as discussed in section 7.2.2. The major disadvantage

is, of course, that we need to keep track of $L$. So we can choose between either a constant memory overhead or a faster algorithm with memory overhead linear in the output.

9.3. **Further improvements.** Algorithm 6 can be improved further, by using the available symmetry in each node. First, for computing MFDS$(R, S)$, the call RFP$(R, \{j\})$ gives the same result for all $j \in [m]$ in the same orbit under the action of Stab$(G, R)$. Therefore, to compute the minimal face-defining set $F \supset R$, we only need to compute *RFP* once for each orbit of $[m] \setminus R$ under the action of Stab$(G, R)$.

**Lemma 15.** *Consider the stabilizer Stab$(G, R) \subset G$ and let $[m] = \bigcup_{i=1}^{t} O_i$ be the orbit decomposition of $[m]$ under the action of the group Stab$(G, R)$. Then for all orbits $i \in [t]$ and any two elements $a, b \in O_i$ we have*

$$RFP(R, \{a\}) = True \qquad \Leftrightarrow \qquad RFP(R, \{b\}) = True.$$

*Proof.* Fix the orbit $i \in [t]$. Let the elements $a, b \in O_i$ and let the permutation $g \in \text{Stab}(G, R)$ such that $a = gb$. Suppose that RFP$(R, \{b\})$ is True. Then there exists a face defining set $F \subset [m]$ such that $R \subset F$ and $b \notin F$. Because $g$ stabilizes $R$, we get $R = gR \subset gF$ and $a = gb \notin gF$. The face defining set $gF$ now implies that RFP$(R, \{a\})$ is True. The implication in the other direction goes analogous. □

We introduce the function MFDS$(R, \text{Stab}(G, R))$ that gives the same output as MFDS$(R)$, but by possibly saving some linear programming computations using Lemma 15. Lemma 16 uses the local symmetry to limit the branching width.

**Lemma 16.** *Consider the subgroup $G(R, S) := (Stab(G, R) \cap Stab(G, S)) \subset G$ and let $J = [m] \setminus \{R, S\}$. If $J = O_1 \cup \ldots \cup O_t$ is the orbit decomposition of $J$ under the action of $G(R, S)$, then for all orbits $i \in [t]$,*

$$FaceEnum(R \cup \{j_i\}, S \cup O_1 \cup \ldots \cup O_{i-1})$$

*enumerates the same face-defining sets up to the action of $G$ for all $j_i \in O_i$.*

*Proof.* Fix the orbit $i \in [t]$. Let the elements $a, b \in O_i$ and let the permutation $g \in G(R, S)$ be such that $a = gb$. Let $F$ be a face-defining set enumerated by

$$\text{FaceEnum}(R \cup \{b\}, S \cup O_1 \cup \ldots \cup O_{i-1}).$$

Then by definition we have:

$$R \cup \{b\} \subset F \qquad \text{and} \qquad F \cap (S \cup O_1 \cup \ldots \cup O_{i-1}) = \emptyset.$$

Applying the action of $g$ to both sides and using the invariances $gR = R$, $gS = S$ and $gO_j = O_j$ for all $j \in [i-1]$ we get:

$$R \cup \{a\} \subset gF \qquad \text{and} \qquad gF \cap (S \cup O_1 \cup \ldots \cup O_{i-1}) = \emptyset$$

and thus $gF$ is by definition enumerated by

$$\text{FaceEnum}(R \cup \{a\}, S \cup O_1 \cup \ldots \cup O_{i-1}).$$

Note that the face-defining sets $gF$ and $F$ represent the same orbit and we can analogous prove the containment in the other direction, so we can conclude our proof. Also note that $a < b$ if and only if $gF \prec F$. □

In particular,

$$\text{FaceEnum}(R \cup \{j_i\}, S \cup O_1 \cup \ldots \cup O_{i-1} \cup \tilde{O}_i)$$

for $j_i \in O_i$ and $\tilde{O}_i \subset O_i$, enumerates only a subset of the face-defining sets that

$$\text{FaceEnum}(R \cup \{j_i\}, S \cup O_1 \cup \ldots \cup O_{i-1})$$

enumerates. So, for every orbit $O_i$ of $J$ under the action of $G(R,S)$, we only need to consider the minimal element of each orbit $O_i$ to recurse on. For the other recursion calls the orbit representatives we find are guaranteed not to be minimal. Lemmas 15 and 16 give us Algorithm 7.

**Input:** $\mathcal{C} = P(A,0)$ and $G \subset \text{Comb}(P)$
**Output:** Minimal representatives for all face-defining set orbits of $P$ under the action of $G$.

**Function** *FaceEnumGroupRecursive(F,S)*:
  **if** $RFP(R,S) = true$ **then**
    $F \leftarrow \text{MFDS}(R, \text{Stab}(G,R))$;
    **if** $\theta_m(F) \neq F$ **then**
      **return**;
    output $F$;
    $J \leftarrow [m] - (F \cup S)$;
    Let $J$ be ordered as $\{j_1, j_2, \ldots, j_t\}$ such that $j_1 < j_2 < \ldots < j_t$;
    Let $J = O_1 \cup \ldots \cup O_{t'}$ the orbit decomposition of $J$ under $G(R,S)$
    such that $O_1 \prec \ldots \prec O_{t'}$;
    **for** $k \leftarrow 1$ **to** $t'$ **do**
      FaceEnumGroupRecursive$(F \cup \{\min O_k\}, S \cup O_1 \cup \ldots \cup O_{k-1})$;
**Function** *FaceEnumGroup()*:
  FaceEnumGroupRecursive$(\emptyset, \emptyset)$

**Algorithm 7:** FaceEnumGroup().

Algorithm 7 can be very useful compared to Algorithm 6, if the stabilizer groups $\text{Stab}(G,R)$ and $G(R,S)$ are large. If the stabilizer $\text{Stab}(G,R)$ is large, computing the minimal face defining set by $\text{MFDS}(R, \text{Stab}(G,R))$ is much faster than by $\text{MFDS}(R)$. Furthermore if $G(R,S)$ splits $J$ in few orbits, the branching width is much smaller. For each orbit $O_i$ of $J$, we save in this way $|O_i| - 1$ calls to MFDS and $|O_i| - 1$ calls to the minimality check.

However, for these improvements, we need to compute the groups $\text{Stab}(G,R)$ and $G(R,S)$. This might cost more than we save, certainly if the obtained groups are small. This is yet to be tested and in practice these improvements might need some heuristics about when to apply them. For example if the sets $F$ and $S$ are under a certain size. Also, if the stabilizer $\text{Stab}(R)$ is already small, we certainly do not need to compute the subgroup $G(R,S) \subset \text{Stab}(R)$.

## 10. A look at the hard dual description problems in dimension 9

In dimension 8, obtaining the dual description of the cone $\mathcal{P}(Q_{E_8})$ was without doubt the hardest part of concluding Voronoi's algorithm. In this section we discuss the two most degenerate cones that appear in Voronoi's algorithm in dimension 9. These cones result from the perfect forms $Q_{129}$ and $Q_{\Lambda_9}$, which we define later. For clarity of the explanation, we work in the equivalent dual setting, i.e. with the Voronoi domains $\mathcal{V}(Q_{E_8}), \mathcal{V}(Q_{129})$ and $\mathcal{V}(Q_{136})$ given by their extreme rays. Remind yourself that $\mathcal{V}(Q) = \text{cone}(\{xx^t : x \in \text{Min}\, Q\})$ is dual to the cone $\mathcal{P}(Q)$.

First we consider a decomposition method, based on Minkowski sums, to relate the facets of a cone to faces of lower dimensional cones. Secondly, we show a relation between the perfect forms $Q_{129}$, $Q_{\Lambda_9}$ and $Q_{E_8}$. Then we use this relation and the decomposition method, to determine the facets of the cone $\mathcal{V}(Q_{129})$, from the facets of the cone $\mathcal{V}(Q_{E_8})$. Lastly we obtain some facts about the cone $\mathcal{V}(Q_{\Lambda_9})$, which could make it feasible to use the Adjacency Decomposition Method to obtain its facets.

10.1. **Conic decomposition method.** Let $\mathcal{C} = \text{cone}(A)$ be a full dimensional cone where $A$ has $m \geq n$ non-redundant column vectors $a_1, \ldots, a_m \in \mathbb{R}^n$. Let $L_1, L_2 \subset [m]$ such that $L_1 \cup L_2 = [m]$ and $L_1 \cap L_2 = \emptyset$. We define $\mathcal{C}_i = \text{cone}(\{a_j : j \in L_i\})$ as the cone spanned by the extreme rays in $L_i$ for $i \in [2]$. Then

$$\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2 := \{a + b : a \in \mathcal{C}_1, b \in \mathcal{C}_2\},$$

also known as the Minkowski sum of $\mathcal{C}_1$ and $\mathcal{C}_2$. Let $n_i := \text{rk}(L_i) := \text{rk}(\text{cone}(\{a_j : j \in L_i\})$ be the rank of the cone $\mathcal{C}_i$. Note that necessarily $n_1 + n_2 \geq n$, because $\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$. Let $\pi_i : \mathbb{R}^n \to \text{span}(\mathcal{C}_i)$ be the orthogonal projection onto the space spanned by $\mathcal{C}_i$ for $i \in [2]$.

We want to determine the faces of the cone $\mathcal{C}$ from faces of the cones $\mathcal{C}_1$ and $\mathcal{C}_2$. Tiwary [Tiw08] showed that this problem is in general NP-hard. Lemma 17 doesn't contradict the hardness, because the number of faces can be exponential in the number of facets and extreme rays.

**Lemma 17.** *If $\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$, then for all face-defining sets $F$ of $\mathcal{C}$ there are face-defining sets $F_1, F_2$ of $\mathcal{C}_1, \mathcal{C}_2$ respectively with $F = F_1 \cup F_2$ and $rk(F_1) + rk(F_2) \geq rk(F)$.*

*Proof.* Suppose that $F \subset [m]$ is a face-defining set of $\mathcal{C}$. Then there exists an $x \in \mathbb{R}^n$ such that

$$\begin{aligned} a_j^t x &= 0 && \forall j \in F \\ a_j^t x &> 0 && \forall j \in [m] \setminus F. \end{aligned}$$

Fix $i \in [2]$ and let $F_i = F \cap L_i$. Then for $\pi_i(x) \in \text{span}(\mathcal{C}_i)$ we have

$$\begin{aligned} a_j^t \pi_i(x) = a_j^t x &= 0 && \forall j \in F_i \\ a_j^t \pi_i(x) = a_j^t x &> 0 && \forall j \in L_i \setminus F_i \end{aligned}$$

and thus $F_i$ is a face-defining set of $\mathcal{C}_i$. Because $L_1 \cup L_2 = [m]$ we can conclude that $F = F_1 \cup F_2$. □

We consider the special case that $\text{rk}(\mathcal{C}_1) + \text{rk}(\mathcal{C}_2) = n_1 + n_2 = n = \text{rk}(\mathcal{C})$. In this case all extreme rays of $\mathcal{C}_1$ are linearly independent from $\mathcal{C}_2$ and visa versa. In particular for face-defining sets $F, F_1$ and $F_2$ of $\mathcal{C}, \mathcal{C}_1$ and $\mathcal{C}_2$ respectively with

$F = F_1 \cup F_2$ we have the equality $\mathrm{rk}(\mathcal{F}) = \mathrm{rk}(F_1) + \mathrm{rk}(F_2)$. See Figure 15 for an example of such a decomposition.

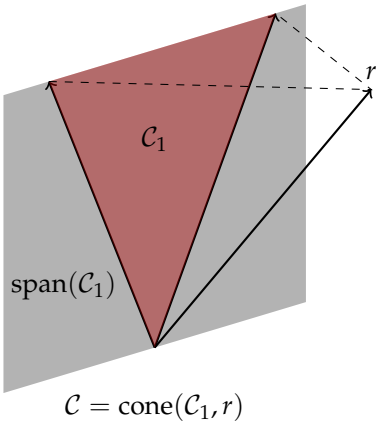


FIGURE 15. Example of a decomposition for which Theorem 18 is applicable. $\mathcal{C}_2 = \mathrm{cone}(r)$ is a rank 1 cone outside of $\mathrm{span}(\mathcal{C}_1)$.

**Theorem 18.** *Suppose $\mathcal{C} = \mathcal{C}_1 + \mathcal{C}_2$ with $rk(\mathcal{C}) = rk(\mathcal{C}_1) + rk(C_2)$. Then the facet-defining sets $F$ of $\mathcal{C}$ are exactly the sets $F_1 \cup L_2$ and $L_1 \cup F_2$ for facet-defining sets $F_1$ and $F_2$ of $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively.*

*Proof.* Let $F$ be a facet-defining set of $\mathcal{C}$, i.e. a face-defining set of $\mathcal{C}$ such that $\mathrm{rk}(F) = n - 1$. Then for the face-defining sets $F_i = F \cap L_i$ of $\mathcal{C}_i$ we have

$$n - 1 = \mathrm{rk}(F) = \mathrm{rk}(F_1) + \mathrm{rk}(F_2) \leq n_1 + n_2 = n.$$

This gives us only two possibilities for $(\mathrm{rk}(F_1), \mathrm{rk}(F_2))$, namely $(n_1 - 1, n_2)$ and $(n_1, n_2 - 1)$. Without loss of generality we only consider the case $(\mathrm{rk}(F_1), \mathrm{rk}(F_2)) = (n_1 - 1, n_2)$. Because $\mathrm{rk}(F_2) = n_2$, the face-defining set $F_2$ of $\mathcal{C}_2$ corresponds to the full cone $\mathcal{C}_2$ and thus $F_2 = L_2$. Also $F_1$ is a facet-defining set of $\mathcal{C}_1$, as $\mathrm{rk}(F_1) = n_1 - 1$. So $F = F_1 \cup L_2$ for some facet-defining set $F_1$ of $\mathcal{C}_1$.

For the converse, let $F_1$ be a facet-defining set of $\mathcal{C}_1$. We must show that $F_1 \cup L_2$ is a facet-defining set of $\mathcal{C}$. Because $F_1$ is a face-defining set of $\mathcal{C}_1$, there exists an $x \in \mathrm{span}(\mathcal{C}_1)$ such that

$$\begin{aligned} a_j^t x &= 0 && \forall j \in F_1 \\ a_j^t x &> 0 && \forall j \in L_1 \setminus F_1. \end{aligned}$$

Consider the orthogonal projection $\pi : \mathrm{span}(\mathcal{C}_2)^\perp \to \mathrm{span}(\mathcal{C}_1)$ onto $\mathrm{span}(\mathcal{C}_1)$. Note that $\mathrm{span}(\mathcal{C}_2)^\perp$ and $\mathrm{span}(\mathcal{C}_1)$ both have dimension $n_1$. Suppose that $\pi(y) = 0$ for some $y \in \mathrm{span}(\mathcal{C}_2)^\perp$. Then $y \in \mathrm{span}(\mathcal{C}_1)^\perp$ and thus

$$y \in \mathrm{span}(\mathcal{C}_1)^\perp \cap \mathrm{span}(\mathcal{C}_2)^\perp \subset \mathrm{span}(\mathcal{C}_1, \mathcal{C}_2)^\perp = (\mathbb{R}^n)^\perp = \{0\}.$$

So the orthogonal projection $\pi$ is injective and thus by a dimension argument gives a bijection between $\mathrm{span}(\mathcal{C}_2)^\perp$ and $\mathrm{span}(\mathcal{C}_1)$. Let $x' := \pi^{-1}(x) \in \mathrm{span}(\mathcal{C}_2)^\perp$. Because $\pi$ is an orthogonal projection onto $\mathrm{span}(\mathcal{C}_1)$ and $x' \in \mathrm{span}(\mathcal{C}_2)^\perp$, we get

that

$$\begin{aligned} a_j^t x' = a_j^t x = 0 \qquad & \forall j \in F_1 \\ a_j^t x' = 0 \qquad & \forall j \in L_2 \\ a_j^t x' = a_j^t x > 0 \qquad & \forall j \in L_1 \setminus F_1. \end{aligned}$$

The existence of such an $x'$ proves that $F_1 \cup L_2$ is a face-defining set of $\mathcal{C}$. Furthermore $\mathrm{rk}(F) = \mathrm{rk}(F_1) + \mathrm{rk}(L_2) = n - 1$ and thus $F_1 \cup L_2$ is a facet-defining set of $\mathcal{C}$. □

10.2. **A general note on two highly degenerate cones.** Finishing Voronoi's algorithm for dimension 8 was not possible for a long time, because of the highly degenerate cone $\mathcal{V}(Q_{E_8})$, belonging to the root lattice $E_8$. $\mathcal{V}(Q_{E_8})$ is a rank 36 cone given by 120 extreme rays. Only in 2005, by exploiting the large automorphism group of $Q_{E_8}$, the dual description of the cone $\mathcal{V}(Q_{E_8})$ was finally found by Dutour Sikirić, et al [DSSV07]: 25.075.566.937.584 facets in 83.092 orbits under the action of $\mathrm{AutMin}(Q_{E_8})$. For dimension 9 we know of 2 perfect forms that give rise to such degenerate cones. Not surprisingly, the corresponding lattices, both consist of shifted layers of the lattice $E_8$. The first one we call $Q_{129}$, as it has $2 \cdot 129$ minimal vectors, and the second one $Q_{\Lambda_9}$ with $2 \cdot 136$ minimal vectors, which corresponds to the laminated lattice $\Lambda_9$ [CS82]. These two perfect forms give rise to rank 45 cones with 129 and 136 extreme rays respectively. Note that, looking at the gap between the number of extreme rays and the dimension, $\mathcal{V}(Q_{129})$ is as degenerate as $\mathcal{V}(Q_{E_8})$ and $\mathcal{V}(Q_{\Lambda_9})$ is even more degenerate. Another problem is that the automorphism groups of these perfect forms are not as large as $\mathrm{Aut}(Q_{E_8})$. Namely, of order 725.760 and 10.321.920 for $\mathrm{Aut}(Q_{129})$ and $\mathrm{Aut}(Q_{\Lambda_9})$ respectively versus the order 696.729.600 for $\mathrm{Aut}(Q_{E_8})$. Therefore, it is a priori not clear that we can use the Adjacency Decomposition Method, which was used for the cone $\mathcal{V}(Q_{E_8})$, to solve the dual description problem of $\mathcal{V}(Q_{129})$ and $\mathcal{V}(Q_{\Lambda_9})$.

As both lattices consist of shifted layers of the lattice $E_8$ the perfect forms $Q_{129}$ and $Q_{\Lambda_9}$ have a lot in common with the perfect form $Q_{E_8}$. In fact, there exist representations of the perfect forms $Q_{E_8}$, $Q_{129}$ and $Q_{\Lambda_9}$, with arithmetical minimum 1, such that

$$Q_{129} = \left[\begin{array}{c|c} Q_{E_8} & b \\ \hline b^t & 1 \end{array}\right] \qquad \text{and} \qquad Q_{\Lambda_9} = \left[\begin{array}{c|c} Q_{E_8} & b' \\ \hline (b')^t & 1 \end{array}\right]$$

for some column vectors $b, b' \in \mathbb{Q}^8$. We can even assume that

$$\mathrm{Min}\, Q_{129} = \{(x,0) : x \in \mathrm{Min}\, Q_{E_8}\} \cup \{\pm(x,1) : x \in S \subset \mathbb{Z}^8 \setminus \{0\} \text{ with } |S| = 9\},$$

$$\mathrm{Min}\, Q_{\Lambda_9} = \{(x,0) : x \in \mathrm{Min}\, Q_{E_8}\} \cup \{\pm(x,1) : x \in S' \subset \mathbb{Z}^8 \setminus \{0\} \text{ with } |S'| = 16\}.$$

The 120 minimal vectors modulo sign originating from $\mathrm{Min}\, Q_{E_8}$ are numbered $1, \ldots, 120$. The remaining 9 and 16 vectors are numbered from 121 up to respectively 129 and 136, depending on which form we work.

10.3. **The cone $\mathcal{V}(Q_{129})$.** For the cone $\mathcal{V}(Q_{129})$ the facets can be obtained from the facets of the cone $\mathcal{V}(Q_{E_8})$, by using Theorem 18.

**Lemma 19.** *The cone $\mathcal{V}(Q_{129})$ has 25.075.566.937.584 + 9 facets.*

*Proof.* We apply Theorem 18 with $L_1 = \{1, \ldots, 120\}$ and $L_2 = \{121, \ldots, 129\}$. Then $n_1 = \mathrm{rk}(L_1) = 36$ and $n_2 = \mathrm{rk}(L_2) = 9$, so together $n_1 + n_2 = 45 = n$. Note that $\mathcal{C}_1$ is exactly an embedding of the cone $\mathcal{V}(Q_{E_8})$ in $\mathrm{span}(\mathcal{C}_1)$ and $\mathcal{C}_2$ is just

a 9-dimensional simplicial cone. So we have 25.075.566.937.584 facets generated by $F_1 \cup L_2$ where $F_1$ is a facet of $\mathcal{V}(Q_{E_8})$ and 9 facets generated by $L_1 \cup S_j$ where $S_j = \{121, \ldots, 129\} \setminus \{j\}$ for $j = 121, \ldots, 129$. □

However, Lemma 19 is not so useful in this form. We do not know all 25.075.566.937.584 facets of $\mathcal{V}(Q_{E_8})$, but only a list of orbit representatives under the action of $\mathrm{AutMin}(E_8)$. To obtain an orbit description of the facets of $\mathcal{V}(Q_{129})$ under the action of $\mathrm{AutMin}(Q_{129})$, we first need to take a closer look at this group.

Consider the automorphism groups $\mathrm{AutMin}(Q_{E_8}) \subset \mathrm{Sym}_{120}$ and $\mathrm{AutMin}(Q_{129}) \subset \mathrm{Sym}_{129}$. They have the following properties:

(1) The orbits of $[129]$ under the action of the group $\mathrm{AutMin}(Q_{129})$ are the two sets $[120]$ and $\{121, \ldots, 129\}$. As a result $\mathrm{Stab}(\mathrm{AutMin}(Q_{129}), [120]) = \mathrm{Stab}(\mathrm{AutMin}(Q_{129}), \{121, \ldots, 129\}) = \mathrm{AutMin}(Q_{129})$.

(2) We consider the action of the group $\mathrm{AutMin}(Q_{129})$ on $[120]$, which is well defined by property (1). This action is equivalent to that of the subgroup $G_{129} \subset \mathrm{AutMin}(Q_{E_8})$ of index 960 in $\mathrm{AutMin}(Q_{E_8})$. We call $G_{129}$ the restriction of $\mathrm{AutMin}(Q_{129})$ to $[120]$. Restricting $\mathrm{AutMin}(Q_{129})$ to $\{121, \ldots, 129\}$ gives a group isomorphic to $\mathrm{Sym}_9$.

**Theorem 20.** *The cone $\mathcal{V}(Q_{129})$ has 25.075.566.937.584 + 9 facets in 71.454.315 + 1 orbits over $\mathrm{AutMin}(Q_{129})$.*

*Proof.* By property (2) the 9 facets of the form $[120] \cup S_i$ lie in the same orbit. By property (1) this orbit is necessarily different from the orbits of the other 25.075.566.937.584 facets. Note that two facets $F_1 \cup \{121, \ldots, 129\}$ and $F_1' \cup \{121, \ldots, 129\}$ lie in the same orbit under the action of $\mathrm{AutMin}(Q_{129})$ if and only if the facets $F_1$ and $F_1'$ of $\mathcal{V}(Q_{E_8})$ lie in the same orbit under the action of $G_{129} \subset \mathrm{AutMin}(Q_{E_8})$. So we apply orbit splitting to convert the 83.092 distinct orbits under the action of $\mathrm{AutMin}(Q_{E_8})$ to orbits under the action of $G_{129}$. For most orbit representatives $F_1$ we have $|\mathrm{Stab}(\mathrm{AutMin}(Q_{E_8}), F_1)| = 1$. So we can use the speedup, mentioned in section 7.1, to split these orbits. The computation resulted in 71.454.315 orbits under the action of $G_{129}$, which are by construction distinct. □

**Corollary 21.** *We have $\mathrm{AutMin}(Q_{E_8}) \times \mathrm{Sym}_9 \subset \mathrm{Comb}(\mathcal{V}(Q_{129}))$.*

*Proof.* This follows immediately from the fact that all elements in $\mathrm{AutMin}(Q_{E_8})$ and $\mathrm{Sym}_9$, acting on $121, \ldots, 129$, act invariantly on the set of facets of $\mathcal{V}(Q_{129})$ found in Lemma 19. □

Corollary 21 is also confirmed by a computation using the Polyhedral GAP package [DS13]. This computation shows that $\mathrm{Lin}_A(\mathcal{V}(Q_{129})) = \mathrm{AutMin}(Q_{E_8}) \times \mathrm{Sym}_9$ with $A = (x_i x_i^t)_{\pm x_i \in \mathrm{Min}\, Q_{129}}$.

10.4. **The cone $\mathcal{V}(Q_{\Lambda_9})$.** We consider the conic decomposition method to obtain the facets of $\mathcal{V}(Q_{\Lambda_9})$. For a similar construction, by Lemma 17, we need not only all 83.092 orbits of facets of the cone $\mathcal{V}(Q_{E_8})$, but also its $(36-2), \ldots, (36-8)$-dimensional faces. Unfortunately the amount of such faces, even under the action of the large automorphism group $\mathrm{AutMin}(Q_{E_8})$, grows extremely fast. A computation showed that the number of orbits of $(36-2)$-faces of $\mathcal{V}(Q_{E_8})$ is already at least $4 \cdot 10^6$. Further experiments show that a lot of these $k$-faces for

$k = 28, \ldots, 34$ do not result in a facet. So there is hope, that the number of orbits of facets of the cone $\mathcal{V}(Q_{\Lambda_9})$ under the action of the automorphism group $\mathrm{AutMin}(Q_{\Lambda_9})$ remains viable.

The order of the group $\mathrm{AutMin}(Q_{\Lambda_9})$ is 5.160.960, instead of the larger group $\mathrm{AutMin}(Q_{E_8})$ of order 348.364.800. Similar to the group $\mathrm{AutMin}(Q_{129})$, the set $[136]$ has 3 orbits under the action of the group $\mathrm{AutMin}(Q_{\Lambda_9})$. Two lie in $[120]$ and the third equals $\{121, \ldots, 136\}$. Restricted to $[120]$ we obtain a subgroup $G_{129} \subset \mathrm{AutMin}(Q_{E_8})$ of index 135 in $\mathrm{AutMin}(Q_{E_8})$.

We characterize the 16 minimal vectors numbered $121, \ldots, 136$ of the perfect form $Q_{\Lambda_9}$ a bit more. There exists an $S \subset \frac{1}{2}\mathbb{Z}^9$ with $|S| = 8$ and a $y \in \frac{1}{2}\mathbb{Z}^9$, such that these 16 minimal vectors are given by

$$\{y \pm x : x \in S\}.$$

We also have $x_i^t Q_{\Lambda_9} y = 0$ for all $i \in [120]$. This characterization is an immediate result from the construction of the lattice $\Lambda_9$ from the lattice $\Lambda_8 = E_8$ [CS82]. As one would expect there exists an automorphism that swaps the minimal vectors $\{y + x : x \in S\} \leftrightarrow \{y - x : x \in S\}$, which is indeed already present in $\mathrm{AutMin}(Q_{\Lambda_9})$. However, the individual swap $y + x \leftrightarrow y - x$ for each $x \in S$ is not present in $\mathrm{AutMin}(Q_{\Lambda_9})$. Luckily it is part of $\mathrm{Comb}(\mathcal{V}(Q_{\Lambda_9}))$, which is easily confirmed without knowing all facets by the fact that they are part of a linear automorphism group of $\mathcal{V}(Q_{\Lambda_9})$. We obtain this from a computation with the Polyhedral GAP package [DS13].

**Fact 22.** *We have the linear automorphism group*

$$\mathit{Lin}_A(\mathcal{V}(Q_{\Lambda_9})) = \langle \mathit{AutMin}(Q_{\Lambda_9}), \{(y + x \leftrightarrow y - x) : x \in S\} \rangle \subset \mathit{Comb}(\mathcal{V}(Q_{\Lambda_9}))$$

*of order* $128 \cdot 5.160.960 = 660.602.880$, *with* $A = (x_i x_i^t)_{x_i \in \mathit{Min}\, Q_{\Lambda_9}}$, *of the cone* $\mathcal{V}(Q_{\Lambda_9})$.

Applying the Adjacency Decomposition Method to obtain the facets of $\mathcal{V}(Q_{\Lambda_9})$ under this larger group seems viable with the adaptations to the Polyhedral software mentioned.

Using the algorithm from Section 9, we were able to compute all orbits of faces of $\mathcal{V}(Q_{\Lambda_9})$ up to some low dimension $k$ under the action of the group from Fact 22. Using this data we can compute whether a permutation might be part of $\mathrm{Comb}(\mathcal{V}(Q_{\Lambda_9}))$ or certainly not. We check this by determining if it leaves the set of $k$-faces invariant. An interesting question is, if more of the symmetry found in $\mathrm{AutMin}(Q_{E_8})$ is present in $\mathrm{Comb}(\mathcal{V}(Q_{\Lambda_9}))$ restricted to $[120]$, instead of only the subgroup of index 135. Although we have not thoroughly checked all possible subgroups in between we can say that index 1 is out of question. We can conclude this, because a generator of $\mathrm{AutMin}(Q_{E_8})$ did not act invariantly on the set of $k$-faces restricted to $[120]$ for $k = 5$.

## 11. Computational results in dimension 9

In this section we will show some of the computational results obtained from running Voronoi's algorithm in dimension 9. By using the improvements discussed in earlier sections, we were able to obtain $\geq$ 23.000.000 perfect 9-dimensional forms. This is significantly more than any earlier published result. We implemented Voronoi's algorithm in the Sage [The17b] software using interfaces to PARI/GP [The17a] and GAP [Gro17]. For GAP we also made extensive use of the in earlier sections mentioned GAP packages Images [JJPW] and Polyhedral [DS13]. Furthermore for dual description of cones with low degeneracy we used the interface of Sage to the Parma Polyhedra Library [BHZ08]. Experiments ran on an Intel Core i7-4790 CPU on a single thread.

For our analysis we want to compute if a perfect form is extreme, semi-eutactic or only perfect. As mentioned in Section 2.7 it is enough to solve a special linear program for each perfect form. We must determine if the optimal value $y$ is strictly positive, zero or strictly negative respectively. The linear program should be solved in exact arithmetic to prove these properties. However, solving such a program in exact arithmetic was too inefficient to apply to all our $\geq$ 23.000.000 perfect forms. Running an exact and inexact solver on the first million forms showed that the error between the two was at most in the order of $10^{-15}$, close to the machine precision of $10^{-16}$. Therefore, we first solved the linear program using inexact arithmetic. Then only if $y \in [-10^{-6}, 10^{-6}]$, we ran the exact solver. As the latter only happened when a form was semi-eutactic, most perfect forms only required a call to the inexact solver. For the inexact and exact solver we used the interface of Sage to GLPK [GNU] and the Parma Polyhedral Library [BHZ08] respectively.

11.1. **Perfect forms.** To find as many perfect forms as efficiently as possible we, after a small initialization time, explored only forms with the minimal number of $2 \cdot 45$ minimal vectors. We explored these perfect forms in the order that they were found. With this set-up we explored around 170.000 perfect forms and found consistently around 1.500.000 new perfect forms each day. In total we have explored 2.672.956 and found 23.638.474 non-similar perfect 9-dimensional forms. So on average we found $\approx 8.84$ new perfect forms for each perfect form we explored. Remarkably, this average did not decay significantly while the algorithm ran, which suggests that we have still only found a small part of the total number of 9-dimensional perfect forms.

As already conjectured by Martinet [Mar02], the number of extreme forms compared to the number of perfect forms seems to decay very fast as the dimension increases. In dimension at most 7 almost all perfect forms were extreme and in dimension 8 only 2408 of the 10916 were extreme. In our exploration so far only 186.089 of the 23.638.474 found perfect forms are extreme; less than 0.8%. Of course, the way we explore could bias this ratio. More on this in Section 11.2.

See Table 1 for an overview of the results of partially running Voronoi's algorithm in dimension 9. Note that the ratio of extreme forms to perfect forms decays fast as the number of minimal vectors decreases. For example on average only 1 out of every 4357 found perfect forms with $2 \cdot 45$ minimal vectors is extreme.

| $\frac{1}{2}\lvert\mathrm{Min}\, Q\rvert$ | Explored | Extreme | Semi-eutactic | Only perfect | Total |
|---|---|---|---|---|---|
| 45 | 2672930 | 1961 | 11 | 8542297 | 8544269 |
| 46 | 19 | 4747 | 19 | 4681725 | 4686491 |
| 47 | 0 | 9299 | 52 | 3952801 | 3962152 |
| 48 | 0 | 14146 | 69 | 2038109 | 2052324 |
| 49 | 0 | 19435 | 109 | 1645783 | 1665327 |
| 50 | 0 | 23488 | 132 | 951485 | 975105 |
| 51 | 0 | 24768 | 117 | 665579 | 690464 |
| 52 | 0 | 23361 | 118 | 400298 | 423777 |
| 53 | 0 | 19393 | 104 | 245611 | 265108 |
| 54 | 0 | 15242 | 74 | 148002 | 163318 |
| 55 | 5 | 11147 | 50 | 78534 | 89731 |
| 56 | 0 | 7422 | 36 | 51353 | 58811 |
| 57 | 0 | 4550 | 33 | 21891 | 26474 |
| 58 | 0 | 2833 | 32 | 13543 | 16408 |
| 59 | 0 | 1714 | 14 | 6374 | 8102 |
| 60 | 0 | 910 | 12 | 3641 | 4563 |
| 61 | 0 | 553 | 10 | 1629 | 2192 |
| 62 | 0 | 376 | 14 | 1321 | 1711 |
| 63 | 0 | 246 | 5 | 390 | 641 |
| 64 | 0 | 156 | 2 | 476 | 634 |
| 65 | 1 | 87 | 2 | 147 | 236 |
| 66 | 0 | 75 | 2 | 126 | 203 |
| 67 | 0 | 47 | 0 | 125 | 172 |
| 68 | 0 | 27 | 0 | 47 | 74 |
| 69 | 0 | 22 | 1 | 21 | 44 |
| 70 | 0 | 21 | 1 | 20 | 42 |
| 71 | 0 | 10 | 0 | 16 | 26 |
| 72 | 1 | 17 | 0 | 4 | 21 |
| 73 | 0 | 4 | 0 | 3 | 7 |
| 74 | 0 | 3 | 0 | 0 | 3 |
| 75 | 0 | 4 | 0 | 0 | 4 |
| 76 | 0 | 3 | 0 | 3 | 6 |
| 77 | 0 | 1 | 0 | 0 | 1 |
| 78 | 0 | 1 | 0 | 0 | 1 |
| 79 | 0 | 1 | 0 | 1 | 2 |
| 80 | 0 | 5 | 1 | 6 | 12 |
| 81 | 0 | 3 | 0 | 0 | 3 |
| 82 | 0 | 3 | 0 | 1 | 4 |
| 84 | 0 | 2 | 0 | 0 | 2 |
| 85 | 0 | 0 | 0 | 2 | 2 |
| 88 | 0 | 0 | 0 | 1 | 1 |
| 90 | 0 | 2 | 0 | 0 | 2 |
| 91 | 0 | 1 | 0 | 0 | 1 |
| 99 | 0 | 1 | 0 | 0 | 1 |
| 129 | 0 | 1 | 0 | 0 | 1 |
| 136 | 0 | 1 | 0 | 0 | 1 |
| Total: | 2672956 | 186089 | 1020 | 23451365 | 23638474 |

TABLE 1. Found perfect forms after partially running Voronoi's algorithm.

11.2. **Extreme forms.** We also ran an adapted version of Voronoi's algorithm that only kept track and explored extreme forms. This was based on the feeling that extreme forms are more likely to be adjacent to extreme forms than non-extreme forms are. We give a heuristic argument for this. Let $Q$ and $Q + R$ be contiguous perfect forms, then $\mathcal{V}(Q)$ is adjacent to $\mathcal{V}(Q + R)$ and $(Q + R)^{-1}$ lies not far from $Q^{-1}$ under the expectation that $R$ is not too large. So by 'being close' it seems more likely that $(Q + R)^{-1} \in \text{Int}(\mathcal{V}(Q + R))$, if $Q^{-1} \in \text{Int}(\mathcal{V}(Q))$, instead of $Q^{-1} \notin \text{Int}(\mathcal{V}(Q))$. In other words, by using Theorem 7, $Q + R$ is more likely to be extreme if $Q$ is extreme.

This feeling seems to be confirmed by the computation of which the results are shown in Table 2. While we only explored 525.278 extreme forms, we found a total of 2.025.641 extreme forms. For comparison, in Section 11.1 we only found 186.089 extreme forms by exploring more than 2 million perfect forms. In particular we found 196.548 extreme forms with $2 \cdot 45$ minimal vectors while before we found only 1961 of those. A necessary note is that there are not enough extreme forms with $2 \cdot 45$ minimal vectors, so the adapted algorithm also explored forms with $2 \cdot 46$ and $2 \cdot 47$ minimal vectors.

11.3. **Estimates on the number of perfect forms.** It is an uncertain business to make any prediction on the number of perfect forms based on a partial exploration. We present some non-rigorous thoughts, based on our current data, to make an estimation. The general absolute upper bound we proved in Section 3 is, of course, not usable to make a precise estimate.

First, we consider the ratio of perfect and extreme forms found while running Voronoi's algorithm and compare this to the number of extreme forms we know by running the adapted version of Voronoi's algorithm. As mentioned earlier only 1 in every 4357 found perfect forms with $2 \cdot 45$ minimal vectors is extreme. But we already found 196.548 of such extreme forms which would heuristically give us at least $4357 \cdot 196.548 \approx 850.000.000$ perfect forms with $2 \cdot 45$ minimal vectors. In the same way we obtain 306.000.000, 148.000.000 and 47.000.000 perfect forms with $2 \cdot 46, 47, 48$ minimal vectors respectively. We can continue this and heuristically get a lower bound of at least 1.400.000.000 non-similar 9-dimensional perfect forms.

Note that the bias for finding extreme forms, mentioned in Section 11.2, could mean that the ratios between perfect and extreme forms found in Table 1 are smaller than on the total set of 9-dimensional perfect forms. If this is the case this lower bound also becomes smaller. The bound would become larger if we find a lot more extreme forms using Voronoi's algorithm restricted to extreme forms. However, because the average number of new extreme forms we find has already dropped significantly, we do not expect there to be many more extreme forms.

Secondly, we consider an upper bound based on the result of Section 10, in particular Theorem 20, that $\mathcal{P}(Q_{129})$ has 71.454.316 orbits of extreme rays under $\text{AutMin}(Q_{129})$. While running the normal variant of Voronoi's algorithm, we checked for 2.393.235 perfect forms with $2 \cdot 45$ minimal vectors, if they were adjacent to any form arithmetically equivalent to $Q_{129}$. Of these 2.393.235 perfect forms exactly 216.798 were adjacent; about 1 in every 11 of such forms.

Suppose we assume that this ratio is the same for all perfect forms, independent of the number of minimal vectors. Because $Q_{129}$ has at most 71.454.316 distinct neighbours up to arithmetical equivalence, we heuristically get an upper bound of approximately 785.000.000 perfect forms. Of course, we have no reason for this

| $\frac{1}{2}\lvert\text{Min } Q\rvert$ | Explored | Extreme |
|---|---|---|
| 45 | 196546 | 196548 |
| 46 | 310199 | 310199 |
| 47 | 18532 | 346872 |
| 48 | 0 | 325702 |
| 49 | 0 | 271100 |
| 50 | 0 | 208295 |
| 51 | 0 | 147417 |
| 52 | 0 | 93875 |
| 53 | 0 | 55617 |
| 54 | 0 | 31329 |
| 55 | 0 | 17399 |
| 56 | 0 | 9386 |
| 57 | 0 | 5019 |
| 58 | 0 | 2843 |
| 59 | 0 | 1645 |
| 60 | 0 | 861 |
| 61 | 0 | 507 |
| 62 | 0 | 334 |
| 63 | 0 | 234 |
| 64 | 0 | 135 |
| 65 | 0 | 79 |
| 66 | 0 | 69 |
| 67 | 0 | 45 |
| 68 | 0 | 27 |
| 69 | 0 | 22 |
| 70 | 0 | 21 |
| 71 | 0 | 10 |
| 72 | 1 | 17 |
| 73 | 0 | 4 |
| 74 | 0 | 3 |
| 75 | 0 | 4 |
| 76 | 0 | 3 |
| 77 | 0 | 1 |
| 78 | 0 | 1 |
| 79 | 0 | 1 |
| 80 | 0 | 3 |
| 81 | 0 | 3 |
| 82 | 0 | 3 |
| 84 | 0 | 2 |
| 90 | 0 | 2 |
| 91 | 0 | 1 |
| 99 | 0 | 1 |
| 129 | 0 | 1 |
| 136 | 0 | 1 |

TABLE 2. Found extreme forms after partially running Voronoi's algorithm with only extreme forms.

ratio to be the same over all perfect forms. However, because the perfect forms with $2 \cdot 45$ minimal vectors seem to take up at least 36% of all perfect forms, this bound is probably not significantly off.

These arguments are far from rigorous as they do not compensate for the bias in the exploration. In particular it stands out that the lower and upper bound contradict each other. Nevertheless, the number of 9-dimensional perfect forms seems to be more in the order of $10^9$, than $10^8$ or $10^{10}$. Because most of these perfect forms are easy to explore, finishing Voronoi's algorithm in dimension 9 seems possible, with the optimizations presented in this thesis. However, we certainly need a heavily parallelised version of Voronoi's algorithm that can run on a supercomputer.

11.4. **A good partitioning function for perfect forms.** If we want to parallelise Voronoi's algorithm over multiple nodes, we need a way to distribute the perfect forms. So we need a good, invariant under arithmetical equivalence, partitioning function

$$f : \{\text{9-dimensional non-similar perfect forms}\} \to \{0, \ldots, k-1\}.$$

With good we mean that it is efficient to compute and gives an almost uniform distribution over the $k$ buckets. Of course, we do not know the full set of 9-dimensional perfect forms, but our computed set of more than 23 million perfect forms probably gives a good indication. To check how our partitioning function $f$ performs we apply it to our dataset and compare the ratio $r$ of the size of largest bucket to the size of the smallest bucket. Optimally we have $r \approx 1$, but we cannot expect such performance in practice. We construct a partitioning function for which we experimentally show that $r \in [1, 1.25]$, if the number of buckets $2 \leq k \leq 1000$ is prime. For $21 \leq k \leq 1000$ with $k$ prime we even have $r \in [1, 1.05]$.

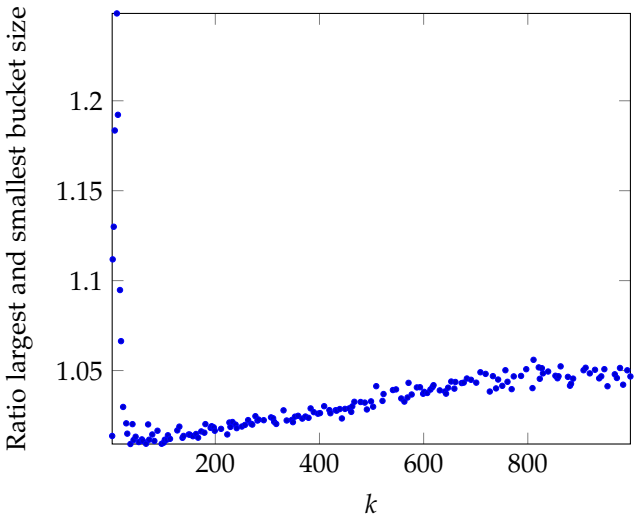


FIGURE 16. Performance of partitioning function $f(Q) = \det(2Q).\text{numerator}()\%k$ for $k$ prime.

The partitioning function we construct is very simple, so it is easy to compute. Let $Q$ be a perfect form with $\lambda(Q) = 1$. Consider the simplified fractional $\frac{a}{b} := \det(2Q)$ with $a, b \in \mathbb{Z}_{>0}$ coprime. Then we set $f(Q) := a \mod k$ with $f(Q) \in \{0, \ldots, k-1\}$. See Figure 16 for the ratio between the largest and smallest bucket

when applying this function on our collection of 23.638.474 non-similar perfect 9-dimensional forms.

Except for very small primes $k \in [2, \ldots, 19]$, the ratio between the size of the largest and the smallest bucket lies in $[1, 1.06]$, which is excellent for such a simple partitioning function. We see that this ratio increases a bit as $k$ grows, but that is expected, as it would also be the case if $f$ is a uniformly random function over $\{0, \ldots, k-1\}$ with only a limited amount of samples. When the number of perfect forms grows, we expect the ratio to become only better.

## References


[AF92] D. Avis and K. Fukuda, *A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra*, Discrete & Computational Geometry **8** (1992), no. 3, 295–313.

[Avi00] D. Avis, *A revised implementation of the reverse search vertex enumeration algorithm*, Polytopes — Combinatorics and Computation, 2000, pp. 177–198.

[BA14] D.M. Brendan and P. Adolfo, *Practical graph isomorphism, II*, Journal of Symbolic Computation **60** (2014), 94 –112.

[Bab16] L. Babai, *Graph isomorphism in quasipolynomial time*, Proceedings of the forty-eighth annual ACM symposium on theory of computing, 2016, pp. 684–697.

[Bac17] R. Bacher, *On the number of perfect lattices*, 2017. working paper or preprint.

[Ban93] W. Banaszczyk, *New bounds in some transference theorems in the geometry of numbers*, Mathematische Annalen **296** (1993), no. 1, 625–635.

[Bar57] E.S. Barnes, *The complete enumeration of extreme senary forms*, Phil. Trans. R. Soc. Lond. A **249** (1957), no. 969, 461–506.

[Bar96] J.L. Baril, *Autour de l'algorithme de Voronoï: construction de réseaux euclidiens*, Ph.D. Thesis, 1996.

[BDSP+14] D. Bremner, M. Dutour Sikirić, D.V. Pasechnik, T. Rehn, and A. Schürmann, *Computing symmetry groups of polyhedra*, LMS Journal of computation and mathematics **17** (2014), no. 1, 565–581.

[BHZ08] R. Bagnara, P.M. Hill, and E. Zaffanella, *The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems*, Science of Computer Programming **72** (2008), no. 1–2, 3–21.

[Bli29] H.F. Blichfeldt, *The minimum value of quadratic forms, and the closest packing of spheres*, Mathematische Annalen **101** (1929), no. 1, 605–608.

[Bli35] ———, *The minimum values of positive quadratic forms in six, seven and eight variables*, Mathematische Zeitschrift **39** (1935), no. 1, 1–15.

[BM00] C. Batut and J. Martinet, *A catalogue of perfect lattices*, 2000. Available at http://jamartin.perso.math.cnrs.fr/Lattices/index.html.

[Che65] N.V. Chernikova, *Algorithm for finding a general formula for the non-negative solutions of a system of linear inequalities*, USSR Computational Mathematics and Mathematical Physics **5** (1965), no. 2, 228–233.

[CK09] H. Cohn and A. Kumar, *Optimality and uniqueness of the leech lattice among lattices*, Annals of Mathematics **170** (2009), no. 3, 1003–1050.

[CKM+17] H. Cohn, A. Kumar, S.D. Miller, D. Radchenko, and M. Viazovska, *The sphere packing problem in dimension 24*, Annals of Mathematics (2017), 1017–1033.

[CR96] T. Christof and G. Reinelt, *Combinatorial optimization and small polytopes*, TOP **4** (1996), no. 1, 1–53.

[CS82] J.H. Conway and N.J.A. Sloane, *Laminated lattices*, Annals of Mathematics (1982), 593–620.

[DFPS00] A. Deza, K. Fukuda, D. Pasechnik, and M. Sato, *On the skeleton of the metric polytope*, Japanese conference on discrete and computational geometry, 2000, pp. 125–136.

[DS13] M. Dutour Sikirić, *GAP Polyhedral package*, 2013.

[DS18] ———, *Algorithm for computing a canonical form of a positive definite quadratic form*, 2018. Available at https://github.com/MathieuDutSik/LattCanonicalize.

[DSSV07] M. Dutour Sikirić, A. Schürmann, and F. Vallentin, *Classification of eight-dimensional perfect forms*, Electronic Research Announcements of the American Mathematical Society **13** (2007), no. 3, 21–32.

[Fej42] L. Fejes, *Über die dichteste kugellagerung*, Mathematische Zeitschrift **48** (1942), no. 1, 676–684.

[FLM97] K. Fukuda, T.M. Liebling, and F. Margot, *Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron*, Computational Geometry **8** (1997), no. 1, 1 –12.

[FP85] U. Fincke and M. Pohst, *Improved methods for calculating vectors of short length in a lattice, including a complexity analysis*, Mathematics of computation **44** (1985), no. 170, 463–471.

[FP96] K. Fukuda and A. Prodon, *Double description method revisited*, Combinatorics and computer science, 1996, pp. 91–111.

[Fuk93] K. Fukuda, *cdd: C-implementation of the double description method for computing all vertices and extremal rays of a convex polyhedron given by a system of linear inequalities*, Department of Mathematics, Swiss Federal Institute of Technology (1993).

[GNU] GNU, *GNU Linear Programming Kit*. Version 4.60.

[Gro17] The GAP Group, *GAP – Groups, Algorithms, and Programming, version 4.8.6*, 2017. Available at https://www.gap-system.org.

[Hal05] T.C. Hales, *A proof of the Kepler conjecture*, Annals of mathematics (2005), 1065–1185.

[HS07] G. Hanrot and D. Stehlé, *Improved analysis of Kannan's shortest lattice vector algorithm*, Annual international cryptology conference, 2007, pp. 170–186.
[JC93] D.O. Jaquet-Chiffelle, *Énumération complete des classes de formes parfaites en dimension 7*, Annales de l'institut Fourier **43** (1993), no. 1, 21–55.
[JJPW18] C. Jefferson, E. Jonauskyte, M. Pfeiffer, and R. Waldecker, *Minimal and canonical images*, Journal of Algebra (2018).
[JJPW] ———, *GAP images package*. Version 1.1.0.
[JK07] T. Junttila and P. Kaski, *Engineering an efficient canonical labeling tool for large and sparse graphs*, Proceedings of the ninth workshop on algorithm engineering and experiments, 2007, pp. 135–149.
[Kep10] J. Kepler, *The six-cornered snowflake*, Paul Dry Books, 2010.
[KZ72] A. Korkine and G. Zolotareff, *Sur les formes quadratiques positives quaternaires*, Mathematische Annalen **5** (1872), no. 4, 581–583.
[KZ73] ———, *Sur les formes quadratiques*, Mathematische Annalen **6** (1873), no. 3, 366–389.
[Laï92] M. Laïhem, *Construction algorithmique de réseaux parfaits*, Ph.D. Thesis, 1992.
[Lin04] S. Linton, *Finding the smallest image of a set*, Proceedings of the 2004 international symposium on symbolic and algebraic computation, 2004, pp. 229–234.
[LLL82] A.K. Lenstra, H.W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Mathematische Annalen **261** (1982), no. 4, 515–534.
[LLS90] J.C. Lagarias, H.W. Lenstra, and C.P. Schnorr, *Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice*, Combinatorica **10** (1990), no. 4, 333–348.
[LV92] H. Le Verge, *A note on Chernikova's algorithm*, Technical Report RR-1662, INRIA, 1992.
[Mar02] J. Martinet, *Perfect lattices in euclidean spaces*, Grundlehren der mathematischen Wissenschaften, Springer Berlin Heidelberg, 2002.
[Mot53] T.S. Motzkin, *The double description method, in contributions to the theory of games II*, Annals of Mathematics Study **28** (1953).
[MV13] D. Micciancio and P. Voulgaris, *A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations*, SIAM Journal on Computing **42** (2013), no. 3, 1364–1391.
[Nam06] Y. Namikawa, *Toroidal compactification of Siegel spaces*, Vol. 812, Springer, 2006.
[Nap96] H. Napias, *Etude expérimentale et algorithmique de réseaux euclidiens*, Ph.D. Thesis, 1996.
[NS06] P.Q. Nguyen and D. Stehlé, *LLL on the average*, International algorithmic number theory symposium, 2006, pp. 238–256.
[PS97] W. Plesken and B. Souvignier, *Computing isometries of lattices*, Journal of Symbolic Computation **24** (1997), no. 3-4, 327–334.
[Rys70] S.S. Ryshkov, *The polyhedron μ(m) and certain extremal problems of the geometry of numbers*, Doklady akademii nauk, 1970, pp. 514–517.
[Sch09] A. Schürmann, *Computational geometry of positive definite quadratic forms: polyhedral reduction theories, algorithms, and applications*, American Mathematical Society, 2009.
[Sco63] P.R. Scott, *On perfect and extreme forms*, Ph.D. Thesis, 1963.
[Sta75] K.C. Stacey, *The enumeration of perfect septenary forms*, Journal of the London Mathematical Society **2** (1975), no. 1, 97–104.
[The17a] The PARI Group, *PARI/GP version 2.9.2*, Univ. Bordeaux, 2017. Available at http://pari.math.u-bordeaux.fr/.
[The17b] The Sage Developers, *SageMath, the Sage Mathematics Software System, version 8.0*, 2017. Available at http://www.sagemath.org.
[Thu10] A. Thue, *Über die dichteste Zusammenstellung von kongruenten Kreisen in einer Ebene*, J. Dybwad, 1910.
[Tiw08] H.R. Tiwary, *On the hardness of computing intersection, union and minkowski sum of polytopes*, Discrete & Computational Geometry **40** (2008), no. 3, 469–479.
[Via17] M.S. Viazovska, *The sphere packing problem in dimension 8*, Annals of Mathematics (2017), 991–1015.
[Voro8] G. Voronoï, *Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs.*, Journal für die reine und angewandte Mathematik **134** (1908), 198–287.
[Wat69] G.L. Watson, *On the minimal points of perfect septenary quadratic forms*, Mathematika **16** (1969), no. 2, 170–177.
[Zie12] G.M. Ziegler, *Lectures on polytopes*, Vol. 152, Springer Science & Business Media, 2012.