Centre de Recherches Mathématiques
CRM Proceedings and Lecture Notes
Volume **48**, 2009

Polyhedral Representation Conversion up to Symmetries

David Bremner, Mathieu Dutour Sikirić, and Achill Schürmann

Abstract. We give a short survey on computational techniques which can be used to solve the representation conversion problem for polyhedra up to symmetries. In particular we discuss decomposition methods, which reduce the problem to a number of lower dimensional subproblems. These methods have been successfully used by different authors in special contexts. Moreover, we sketch an incremental method, which is a generalization of Fourier–Motzkin elimination, and we give some ideas how symmetry can be exploited using pivots.

1. Introduction

By the Farkas–Minkowski–Weyl Theorem a convex polyhedron in $\mathbb{R}^d$ has two representations. It can either be described by a finite set of linear inequalities (facets) or by a finite set of generators (vertices and rays). Precise definitions are given in Section 2.

One of the most fundamental problems in the theory of polyhedra and its applications, such as Combinatorial Optimization or Computational Geometry, is the conversion between two different descriptions. Many algorithms for this representation conversion have been proposed (see for example [51, 4, 11, 43]). For certain classes of polyhedra efficient methods are known, but there is no known approach which efficiently solves the problem in general. Programs like `cdd` [33], `lrs` [2], `pd` [52], `porta` [17] (or `polymake` [36] either relying on some of the others or using its own method closely related to `cdd`) allow conversion of the representation of a polyhedron. Since these programs are implementations of quite different methods, their efficiency may vary tremendously on a given example.

2000 *Mathematics Subject Classification.* 68W05.

The first author acknowledges the support of the Natural Sciences and Engineering Research Council of Canada and the Alexander von Humboldt Foundation. The second author has been supported by the Croatian Ministry of Science, Education and Sport under contract 098-0982705-2707. The third author was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant SCHU 1503/4-1.

The authors would like to acknowledge fruitful discussions with David Avis, Antoine Deza, Komei Fukuda, Alexander Hulpke, Michael Joswig, Jesus de Loera, Brendan McKay, Hugh Thomas, and Frank Vallentin. They would also like to thank the Centre de recherches mathématiques for making possible the workshop at which many of these discussions took place.

This is the final form of the paper.

Many interesting polyhedra both pose difficulties for the standard representation conversion approaches and have many symmetries that could potentially be exploited. In many applications it is sufficient (or at least necessary) to obtain a list of inequalities or generators up to symmetries. In the present paper we give a brief survey of approaches that can be used for the representation conversion problem up to symmetries. We do not discuss their asymptotic complexity (which, in the worst case, is not encouraging), but rather refer to previous papers where the methods have proven themselves on difficult instances that could not otherwise be solved. For the new approach discussed in Section 7 we provide some experimental data ourselves.

The paper is organized as follows. In Section 2 we give some basic notations and facts from the theory of convex polyhedra. In Section 3 we consider different notions of symmetries and describe how they can practically be obtained using a graph automorphism computation. In Section 4 we describe the group theoretical notions used in the representation conversion methods discussed in the remainder of the paper. In Section 5 we consider decomposition methods which reduce the given problem to a number of smaller problems. These approaches have been used quite successfully by different authors. In Section 6 we describe the incremental *Cascade algorithm* and in Section 7 we show how symmetry can be exploited in a simplex pivot based algorithm.

2. Convex polyhedra

In this section, we give a brief introduction to some basic concepts and terminology of (convex) polyhedra. For more details on polyhedra, we refer to the books [65, 38, 57].

Given the vector space $\mathbb{R}^d$, denote by $(\mathbb{R}^d)^*$ its dual vector space, i.e. the vector space of linear functionals on $\mathbb{R}^d$. A *convex polyhedron* $\mathcal{P} \subseteq \mathbb{R}^d$ can be defined by a finite set of linear inequalities

$$\mathcal{P} = \{x \in \mathbb{R}^d : f_i(x) \geq b_i, i = 1, \ldots, m\}$$

with $f_i \in (\mathbb{R}^d)^*$ and $b_i \in \mathbb{R}$ for $i = 1, \ldots, m$. If the number of inequalities m in the description is minimum, we speak of a non-redundant description. The dimension $\dim \mathcal{P}$ of $\mathcal{P}$ is the dimension of the smallest affine subspace containing it. Under the assumption that $\mathcal{P}$ is full-dimensional, i.e. $\dim \mathcal{P} = d$, every inequality i of a non-redundant description defines a *facet* $\{x \in \mathcal{P} : f_i(x) = b_i\}$ of $\mathcal{P}$, which is a $(d-1)$-dimensional convex polyhedron contained in the boundary of $\mathcal{P}$.

By the Farkas–Minkowski–Weyl Theorem (see e.g. [57, Corollary 7.1a]), $\mathcal{P}$ can also be described by a finite set of generators:

$$\begin{aligned}\mathcal{P} &= \operatorname{conv}\{v_1, \ldots, v_k\} + \operatorname{cone}\{v_{k+1}, \ldots, v_n\} \\ &= \left\{ \sum_{i=1}^{n} \lambda_i v_i : \lambda_i \geq 0, \sum_{i=1}^{k} \lambda_i = 1 \right\}\end{aligned}$$

where $v_i \in \mathbb{R}^d$ for $i = 1, \ldots, n$. If the number of generators is minimum, the description is again called *non-redundant*. In the non-redundant case, the generators v_i, $i = 1, \ldots, k$, are called *vertices* and $\mathbb{R}_{\geq 0} v_i$, $i = k+1, \ldots, n$, are the *extreme rays* of $\mathcal{P}$. In case $\mathcal{P}$ is bounded we have $n = k$ and we speak of a *convex polytope*.

The *representation conversion* from a minimal set of generators into a minimal set of linear functionals (or vice versa) is called the *dual description problem*. By

using homogeneous coordinates, the general inhomogeneous problem stated above can be reduced to the homogeneous one where $b_i = 0$ and $k = 0$. For example, we embed $\mathcal{P} \in \mathbb{R}^d$ in the hyperplane $x_{d+1} = 1$ in $\mathbb{R}^{d+1}$ and consider the closure of its conic hull. The so-obtained polyhedron $\mathcal{P}' = \text{cone}\{v'_1, \ldots, v'_n\}$, with $v'_i = (v_i, 1)$ for $i = 1, \ldots, k$ and $v'_i = (v_i, 0)$ for $i = k+1, \ldots n$, is referred to as *polyhedral cone*.

By duality, the problem of converting a description by homogeneous inequalities into a description by extreme rays is equivalent to the opposite conversion problem. So for simplicity we assume from now on that $\mathcal{P} \subseteq \mathbb{R}^d$ is a *polyhedral cone* given by a minimal (non-redundant) set of generators (extreme rays). If

$$\mathcal{P} = \text{cone}\{v_1, \ldots, v_n\}$$

we say that $\mathcal{P}$ is generated by $v_1, \ldots, v_n \in \mathbb{R}^d$.

We want to find a minimal set $\{f_1, \ldots, f_m\} \subset (\mathbb{R}^d)^*$ with

$$\mathcal{P} = \{x \in \mathbb{R}^d : f_i(x) \geq 0, i = 1, \ldots, m\}.$$

By choosing a suitable projection, it is possible to reduce the problem further to the case where $\mathcal{P}$ is full-dimensional and does not contain any non-trivial linear subspaces. For example, if $v_1, \ldots, v_n \in \mathbb{R}^d$ span a k-dimensional linear subspace, we may just choose (project onto) k independent coordinates. The appropriate projection (i.e. equations of the linearity space) can be found efficiently via Gaussian elimination. In other words, without loss of generality we may assume the v_i span $\mathbb{R}^d$ ($\mathcal{P}$ is full-dimensional) and the linear inequalities f_i span $(\mathbb{R}^d)^*$ ($\mathcal{P}$ is pointed).

A *face* of $\mathcal{P}$ is a set $\{x \in \mathcal{P} : f(x) = 0\}$ where f is an element of

$$\mathcal{P}^* = \{f \in (\mathbb{R}^d)^* : f(x) \geq 0 \text{ for all } x \in \mathcal{P}\},$$

the polyhedral cone *dual* to $\mathcal{P}$. Note that $(\mathcal{P}^*)^* = \mathcal{P}$. The faces of a pointed polyhedral cone are themselves pointed polyhedral cones. We speak of a k-face, if its dimension is k. The faces form a (combinatorial) lattice ordered by inclusion, the *face lattice* of $\mathcal{P}$. The rank of a face in the lattice is given by its dimension. Each face is generated by a subset of the generators of $\mathcal{P}$ and therefore it is uniquely identified by some subset of $\{1, \ldots, n\}$. In particular, the 0-dimensional face is identified with the empty set $\emptyset$ and $\mathcal{P}$ itself with the full index set $\{1, \ldots, n\}$. All other faces of $\mathcal{P}$ are identified with some strict, non-empty subset $F \subset \{1, \ldots, n\}$.

We write $F \lhd F'$ for two faces of $\mathcal{P}$ with $F \subset F'$ and $\dim F = \dim F' - 1$. Two k-faces of $\mathcal{P}$ are said to be *adjacent*, if they contain a common $(k-1)$-face and are contained in a common $(k+1)$-face. In particular, two extreme rays (1-faces) are adjacent, if they generate a common 2-face and two facets are adjacent, if they share a common $(d-2)$-face (a *ridge*). By the properties of a lattice, for two faces F_1 and F_2, not necessarily of the same dimension, there is always a unique largest common face $F = F_1 \cap F_2$ contained in them, and a unique smallest face F', containing both. Any sub-lattice $[F : F']$ of the face lattice consisting of all faces containing F and contained in F' is known to be isomorphic to the face lattice of some pointed polyhedral cone of dimension $\dim F' - \dim F$. Note that this is a special feature of face lattices of polyhedra, which is not true for general lattices. In particular the *diamond property* holds for face lattices: Every sub-lattice of rank 2 carries the combinatorics of a 2-dimensional polyhedral cone, namely a face F of rank $k-1$, a face F' of rank $k+1$ and two faces F_1 and F_2 of rank k. The rank 2 sub-lattices are in one-to-one correspondence to pairs of adjacent faces.

A *polyhedral complex* Δ is a set of polyhedral cones (the *cells* of Δ) satisfying the following two properties:

(a) If $\mathcal{P} \in \Delta$ then every face of $\mathcal{P}$ is also in Δ.
(b) For all $\mathcal{P}_i, \mathcal{P}_j \in \Delta$, $\mathcal{P}_i \cap \mathcal{P}_j$ is a face of both $\mathcal{P}_i$ and $\mathcal{P}_j$.

The facets of a polyhedral cone $\mathcal{P}$ form a polyhedral complex, the *boundary complex* of $\mathcal{P}$. Polyhedral complex Δ' is a *subdivision* of polyhedral complex Δ if every cell of Δ is the union of cells in Δ' and every cell of Δ' is contained in some cell of Δ. A subdivision is called a *triangulation* if every cell is a simplicial cone, i.e. is the conic hull of exactly d-extreme rays. By the homogeneous embedding of polytopes discussed above, we may equally discuss boundary complexes, subdivisions, and triangulations for polytopes. Two faces (cells) of a polytopal complex have empty intersection exactly when the corresponding cones intersect only at the origin.

3. Polyhedral symmetries

3.1. Groups acting on polyhedra. In this paper we are especially interested in the case where some non-trivial group acts on the given polyhedron $\mathcal{P}$. The *combinatorial automorphism group* of $\mathcal{P}$, generated by n extreme rays, is the subgroup of $\mathrm{Sym}(n)$ of all permutations (acting on ray indices $\{1, \ldots, n\}$) which preserve the whole face lattice of $\mathcal{P}$. Thus the combinatorial automorphism group acts not only on the generating rays, but also on the set of facets (inequalities), and more generally on the sets of faces of a given dimension. Moreover, it preserves the inclusion relation among faces. It is known that the combinatorial automorphism group of $\mathcal{P}$ can be computed from the incidence relations between extreme rays and facets (see [44]).

Given generators $v_1, \ldots, v_n \in \mathbb{R}^d$ of a polyhedral cone $\mathcal{P}$ and a subgroup G of the combinatorial automorphism group of $\mathcal{P}$, we want to obtain the facets of $\mathcal{P}$ up to symmetry, that is, one representative $F \subset \{1, \ldots, n\}$ from each orbit under the action of G.

If no group or only a small group is given, we usually still want to exploit as much symmetries of the given polyhedron as possible in the representation conversion. That is, ideally we would like to compute the full combinatorial automorphism group. However, we do not know how to compute it without computing the facets, which is precisely the problem we want to solve. So we have to settle for a compromise and work with a more restricted type of automorphism. Note though, that one may obtain the full combinatorial automorphism group of $\mathcal{P}$, after the facets have been computed

In many cases the combinatorial automorphism group (or some nontrivial subgroup) may reflect geometric symmetries, for example if it has a representation as a subgroup of $\mathrm{GL_d}(\mathbb{R})$ acting naturally on $\mathbb{R}^d$ and the polyhedral cone $\mathcal{P}$. The group of all matrices in $\mathrm{GL_d}(\mathbb{R})$ preserving $\mathcal{P}$ is called *linear automorphism group* of $\mathcal{P}$. Since a linear automorphism permutes the set of extreme rays $\{\mathbb{R}_{\geq 0}v_1, \ldots, \mathbb{R}_{\geq 0}v_n\}$, we naturally obtain a representation as a permutation group $G \leq \mathrm{Sym}(n)$. It is important to note here that although the induced permutation group is finite, the linear automorphism group of $\mathcal{P}$ is not necessarily so, and it can be quite awkward. Think for example of the quadrant in $\mathbb{R}^2$ generated by the two non-negative coordinate axes. The induced permutation group is $\mathrm{Sym}(2)$, its linear automorphism

group however is

$$\left\{ \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}, \begin{pmatrix} 0 & c \\ d & 0 \end{pmatrix} : a, b, c, d \in \mathbb{R}_{>0} \right\}.$$

As explained in Section 2 we may limit our discussion to the special case of a full-dimensional, pointed polyhedral cone $\mathcal{P}$ generated by $v_1, \ldots, v_n \in \mathbb{R}^d$. The elements $A \in \mathrm{GL_d}(\mathbb{R})$ of the linear automorphism group of $\mathcal{P}$ satisfy $Av_i = \lambda_i v_{\sigma(i)}$, where $\sigma \in \mathrm{Sym}(n)$ is an induced permutation and $\lambda_i > 0$. Note, in case $\mathcal{P}$ is the homogenization of a $(d-1)$-dimensional polyhedron $\mathcal{P}'$, the linear automorphism group of $\mathcal{P}$ corresponds to the so called *projective automorphism group* of $\mathcal{P}'$, that is, the set of all projective maps preserving $\mathcal{P}'$. We are not aware of any practical algorithm to compute linear (or projective) automorphism groups or to decide if two polyhedral cones are linear (or projective) isomorphic.

3.2. Restricted isomorphisms. A *restricted isomorphism* of two full-dimensional vector families $V = \{v_1, \ldots, v_n\}$ and $V' = \{v'_1, \ldots, v'_n\}$ in $\mathbb{R}^d$ is given by a matrix $A \in \mathrm{GL_d}(\mathbb{R})$ such that there exists a permutation σ satisfying $Av_i = v'_{\sigma(i)}$ for $i = 1, \ldots, n$. A *restricted automorphism* of a vector family is a restricted isomorphism of V with itself.

We speak of a restricted isomorphism between two polyhedral cones generated by the two vector families V and V', if it is a restricted isomorphism between V and V'. Note though that the definition of restricted isomorphisms for polyhedral cones depends strongly on the choice of generators. Since such generators are only unique up to any positive factor, the choice of generators is very crucial. In practice the situation is usually not so bad as there is often a natural choice for the generators.

The big advantage of restricted isomorphisms is that we can test their existence and compute them by solving the same questions for edge coloured graphs. Given a vector family $V = \{v_1, \ldots, v_n\} \subset \mathbb{R}^d$ spanning $\mathbb{R}^d$ we define the positive definite matrix

$$Q = \sum_{i=1}^{n} v_i v_i^t. \tag{3.1}$$

Let the graph $G(V)$ be the complete graph with vertices v_i and edge colours $c_{ij} = v_i^t Q^{-1} v_j$. Then the following holds:

Proposition 3.1. *Let $V, V' \subset \mathbb{R}^d$ be two finite vector families. Then every isomorphism of the edge coloured graphs $G(V)$ and $G(V')$ yields a restricted isomorphism of V and V' and vice versa.*

In practice the popular and very nice program `nauty` [53] by McKay can be used to check for graph isomorphisms or compute the group of automorphisms. Note however, that the current version only takes vertex coloured graphs as input. We therefore have to transform our edge coloured graph to a somewhat larger vertex coloured graph that preserves the automorphism group (see for example [54, p. 25]).

Proof of Proposition 3.1. Let Q be the matrix (3.1) obtained from V. Denote by R the unique square root of Q^{-1}, that is, the positive definite $d \times d$ matrix R with $Q^{-1} = R^2$. Let $w_i = Rv_i$ for $i = 1, \ldots, n$. Then the edge colours c_{ij} of the graph $G(V)$ are exactly the inner products $w_i^t w_j$ of the transformed vectors w_i and w_j. In the same way we obtain Q', R' and w'_i for V'.

Now let $A \in \mathrm{GL}_d(\mathbb{R})$ be a restricted isomorphism for V and V' with associated permutation $\sigma \in \mathrm{Sym}(n)$. Then we have $AQA^t = Q'$. Moreover, the matrix $T = R'AR^{-1}$ is orthogonal and satisfies $Tw_i = w'_{\sigma(i)}$. This implies $c_{ij} = c'_{\sigma(i)\sigma(j)}$, i.e. the restricted isomorphism of the vector families V and V' corresponds to an isomorphism between the edge coloured graphs $G(V)$ and $G(V')$.

Suppose on the other hand that $\sigma \in \mathrm{Sym}(n)$ is an isomorphism between the edge coloured graphs $G(V)$ and $G(V')$. By reordering elements, we may simply assume $\sigma = \mathrm{Id}$. Instead of looking for a solution of $Av_i = v'_i$ we consider the equivalent equations $Tw_i = w'_i$. Since the v_i generate $\mathbb{R}^d$, we find a basis $(v_{i_1}, \ldots, v_{i_d})$ of $\mathbb{R}^d$. If P, respectively P' is the $d \times d$ matrix formed by (w_{i_k}), respectively (w'_{i_k}), then the matrix equation $(c_{ij}) = (c'_{ij})$ takes the form $P^tP = P'^tP'$. So, the matrix $T = P'P^{-1}$ is orthogonal and we have for any $k = 1, \ldots, d$ and $j = 1, \ldots, n$:

$$w'^t_{i_k} T w_j = (Tw_{i_k})^t T w_j = w^t_{i_k} w_j = w'^t_{i_k} w'_j.$$

This yields $w'^t_i(Tw_j - w'_j) = 0$ and since the w'_i form a basis of $\mathbb{R}^d$ the relations $Tw_j = w'_j$ are implied. Thus we obtain a restricted isomorphism between V and V'. □

4. Orbits of faces

4.1. Dealing with orbits. In order to generate facets (or more generally faces) of a polyhedron up to symmetries, it is necessary to deal with orbits of faces. In this section we briefly indicate the basic tasks that we have to accomplish and how these can be approached.

As before, we assume that a polyhedral cone $\mathcal{P}$ is given by a set of generators $v_1, \ldots, v_n \in \mathbb{R}^d$ which define the extreme rays of $\mathcal{P}$. Each face is represented by a subset of $\{1, \ldots, n\}$ which corresponds to the indices of generators incident to the face. We assume $G \leq \mathrm{Sym}(n)$ is some subgroup of the combinatorial automorphism group of $\mathcal{P}$, hence a permutation group acting not only on the set of indices $\{1, \ldots, n\}$ (respectively rays), but also on the whole face lattice of $\mathcal{P}$. In particular, the dimension of a face and inclusion between faces are preserved by every group element.

In the most general form, the problem we have to solve is the following: Given two subgroups G_1 and G_2 of $\mathrm{Sym}(n)$ and a list L_1 of G_1-inequivalent faces, we need to obtain a list L_2 of G_2-inequivalent faces.

For example, if $G_1 = \{\mathrm{Id}\}$ is trivial, the list $L_1 = \{F_1, \ldots, F_k\}$ simply is a list of pairwise unequal faces and we may obtain L_2 by testing for G_2-equivalence: Starting with $L_2 = \{F_1\}$ and then subsequently adding F_i for $i = 2, \ldots, n$ to L_2, if it is not G_2-equivalent to any element in L_2. Clearly, this *orbit fusion* can be applied whenever $G_1 \leq G_2$.

In case $G_2 < G_1$ it is necessary to "break some symmetry" and *split* (*factorize*) *orbits*. This can be done with the *double coset decomposition*, described in Section 4.2. Note, if neither $G_1 \leq G_2$ nor $G_2 < G_1$, one could in principle convert the G_1-orbits into G_2-orbits in two steps, by either using the intersection group $G = G_1 \cap G_2$ or the group $G = \langle G_1, G_2 \rangle$ generated by G_1 and G_2 and obtaining G-orbits in an intermediate step.

Orbit fusing and splitting is a typical and essential task when generating discrete structures up to isomorphism (see for example [39, 46, 12, 45]).

4.2. Fusing orbits, equivalence tests and canonical representatives. A common task is to decide whether or not two faces are G-equivalent, that is, whether or not the corresponding subsets of $\{1,\dots,n\}$ lie in the same orbit under the action of G.

The dimension of a face (rank in the face lattice) and its cardinality are quickly testable invariants. Other invariants can easily be obtained, for example by looking at the action of G on pairs, triples, or other k-tuples of indices (generators), respectively on lower dimensional faces. The number of elements from each such orbit included in a face is a G-invariant. Unfortunately, there is no clear rule how many such sets have to be chosen, so one has to rely on heuristics.

If we only consider restricted isomorphisms, metric invariants can be used as well, for example the set of pairwise inner products between generators, discussed further in Section 7.4.2.

When all invariants are satisfied, then group computations must be done. For small groups it is possible to simply generate the whole orbit of a face. If the size of available memory is a problem, we may just keep a *canonical representative* for the orbit, for example the lexicographical minimum (when viewed as a subset of $\{1,\dots,n\}$). These can be found for example by a backtrack method (see Section 4.4).

4.3. Breaking symmetry by splitting orbits using double cosets. Assume $G_2 < G_1 \leq \mathrm{Sym}(n)$ and that a list of G_1-orbits is given. In order to obtain a list of G_2- orbits, we may split each orbit G_1F, with a representative face F, by the well known *double coset decomposition* (cf. [12, 46]): The group G_1 can be decomposed into *double cosets*

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

where $g_1,\dots,g_r$ are elements of G_1. Then the orbit G_1F is decomposed into

$$G_1F = \bigcup_{i=1}^{r} G_2 g_i F,$$

hence into r orbits G_2F_i with $F_i = g_iF$.

Note that the more straightforward algorithm of generating the full orbit or of computing a decomposition of G_1 into cosets G_2g_i is slower and/or requires more memory (see [46]).

4.4. Data structures and implementation issues. The fundamental data structures used to work with permutation groups are *bases and strong generating sets* (BSGS) (see for example [59, 40]). Based on them a backtrack search on cosets of a point stabilizer chain can be used to obtain canonical representatives, to decide on (non-)equivalence as well as to obtain stabilizers of subsets of $\{1,\dots,n\}$ (faces). For details we refer to [59, 45]. An elaborate version is the *partition backtrack* introduced by Leon [48] (cf. [49, 59]). These methods are known to work quite well in practice, although from a complexity point of view the problems are known to be difficult. That is, there is no (worst-case) polynomial time algorithm known to solve these problems. Even worse, it is somewhat unlikely that there exist polynomial time algorithms, since the *graph isomorphism problem*, not known to be in P, is reducible to them in polynomial time (cf. [50]).

The computer algebra system GAP [35] provides functions for generation of full orbits (Orbit) stabilizer computations (Stabilizer) and equivalence tests (RepresentativeAction).

In the case where we have special knowledge of the group G or its representation, it might be much easier to obtain canonical representatives or to compute stabilizers of faces. For example, we may have a situation where the symmetric group $\mathrm{Sym}(n)$ acts on n elements (see [1, 13]). Another example where the action of the symmetric group is used is described in [27] (see [26] for corresponding computer code). Typically, polyhedra arising in Combinatorial Optimization are convex hulls of (0/1)-vectors, where each coordinate (variable) represents an edge of a complete directed or undirected graph with n vertices on which $\mathrm{Sym}(n)$ acts. In [18] a method for obtaining canonical representatives in this situation is described.

5. Decomposition methods

In this section we describe two methods which reduce the facet generation problem under symmetries to a number of smaller instances of facet generation problems. In contrast to the original problem, solving the smaller problems (for subcones) may be feasible for available software, such as `cdd` or `lrs`. These techniques have been proven to be successful in practice, in cases where standard approaches failed.

We mainly distinguish two approaches, the *Incidence Decomposition Method* (see Section 5.1) and the *Adjacency Decomposition Method* (see Section 5.2). Both methods are reasonably natural and have been used separately by different authors.

5.1. Incidence decomposition method. The Incidence Decomposition Method reduces the problem of facet generation to a number of smaller problems, in which we generate facets that are incident to some extreme rays. As before let $\mathcal{P}$ be a polyhedral cone in $\mathbb{R}^d$, generated by $v_1, \ldots, v_n$. Assume $G \leq \mathrm{Sym}(n)$ is some permutation group acting on the face lattice of $\mathcal{P}$. The set of extreme rays (indices) falls into orbits under the action of G. For each orbit we consider a representative r_i (index i) and generate a list of G-inequivalent facets of $\mathcal{P}$ incident to it. Then, in a post-processing step the lists of facets obtained in this way are merged to a list of G-inequivalent facets of $\mathcal{P}$ (see Section 4). Since every G-orbit of facets of $\mathcal{P}$ contains a facet which is incident to one of the chosen representatives, the resulting list is complete.

Input: n extreme rays of a polyhedral cone $\mathcal{P}$ and a group $G \leq \mathrm{Sym}(n)$ acting on $\mathcal{P}$'s face lattice.
Output: complete set $\mathcal{F}$ of G-inequivalent facets of $\mathcal{P}$.

```
F ← ∅.
R ← complete set of G-inequivalent extreme rays of P.
for r ∈ R do
  F_r ← facets of P incident to r.
  for F ∈ F_r do
    if F is G-inequivalent to facets in F then
      F ← F ∪ {F}.
    end if
  end for
end for
```

The main computational gain comes from the following: when we compute the facets that are incident to a given ray r_i, we may not have to consider all n extreme rays of $\mathcal{P}$, because some of the rays may not be adjacent to r_i. These are exactly the extreme rays $r_j = \mathbb{R}_{\geq 0} v_j$ which give redundant inequalities of the polyhedron

$$\mathcal{P}_i^* := \{f \in (\mathbb{R}^d)^* : f(v_j) \geq 0, j = 1, \ldots, n \text{ and } f(v_i) = 0\}.$$

So, for each of the $n-1$ rays with index in $\{1, \ldots, n\} \setminus \{i\}$ we may solve a linear program in $(\mathbb{R}^d)^*$ to decide redundancy. As an outcome we obtain a list of $n' < n$ rays needed in the definition of $\mathcal{P}_i^*$. The smaller n', the bigger the computational gain. Note, that $\mathcal{P}_i^*$ is of dimension $d-1$. Its dual contains the line $\{\lambda v_i : \lambda \in \mathbb{R}\}$ and we may project it (or at least think of it as projected) down along this line to a $d-1$ dimensional polyhedral cone.

So the problem of enumerating facets incident to a given ray is a facet enumeration problem, but in one lower dimension and with fewer extreme rays. Some of the lower dimensional subproblems may still be too difficult; in this case we may apply the method recursively. We come back to this in Section 5.3.

The Incidence Decomposition Method has been used in [37] for finding the vertices of the metric polytope MET_7. The method was also introduced in [19] along with the Adjacency Decomposition Method but it was found to be less competitive for their application. The Incidence Decomposition Method was also discussed by Fukuda and Prodon [34].

5.2. Adjacency decomposition method. As with the Incidence Decomposition Method, the Adjacency Decomposition Method is a reasonably natural method for computing the facets of a polytope up to symmetries. So it is no wonder that the method was discovered several times, for example, in [41] as "algorithm de l'explorateur," in [19] as "adjacency decomposition method" and in [25] as "subpolytope algorithm." Other example of applications are in [32, 22] or [23]. The adjacency decomposition scheme is also implicit in polyhedral decomposition schemes, i.e. when a space is decomposed as an union of polyhedral cones. Examples are in [60, 62, 63] (cf. [30, 29]). Rather than focusing on the incidence of (orbits of) facets to extreme rays as the Incidence Decomposition Method does, the Adjacency Decomposition Method focusses on the incidence of facets with other facets. Starting from a (set of) initial G-inequivalent facets, it traverses the adjacency graph of facet orbits. The initial facet(s) may be obtained by suitable linear programs, or in strongly polynomial time using the methods described in [11].

Input: n extreme rays of a polyhedral cone $\mathcal{P}$ and a group $G \leq \mathrm{Sym}(n)$ acting on $\mathcal{P}$'s face lattice.
Output: complete set $\mathcal{F}$ of G-inequivalent facets of $\mathcal{P}$.

```
T ← {F} with F a facet of P.
F ← ∅.
while there is an F ∈ T do
    F ← F ∪ {F}.
    T ← T \ {F}.
    H ← facets of F.
    for H ∈ H do
        F' ← facet of P adjacent to F along H.
        if F' is G-inequivalent to all facets in F ∪ T then
            T ← T ∪ {F'}.
```

end if
end for
end while

The facet F' of $\mathcal{P}$ with $F \cap F' = H$ for a given ridge $H \lhd F$ can be found by a *gift-wrapping step* (cf. [14, 61]). Let $v_1, \ldots, v_n$ be the generators of $\mathcal{P}$'s extreme rays. The defining inequality $f \in (\mathbb{R}^d)^*$ of the facet F' should satisfy $f(v_i) = 0$ for all generators $v_i \in H$. The vector space of such functions has dimension 2. Let us select a basis $\{f_1, f_2\}$ of it. If $f = \alpha_1 f_1 + \alpha_2 f_2$ is the defining inequality of F, H or F', then $f(v_i) \geq 0$ for all $i \in \{1, \ldots, n\}$. This translates into a set of linear inequalities on α_1, α_2 defining a 2-dimensional pointed polyhedral cone. One finds easily its two generators $(\alpha_1^i, \alpha_2^i)_{1 \leq i \leq 2}$. The corresponding inequalities $f_i = \alpha_1^i f_1 + \alpha_2^i f_2 \in (\mathbb{R}^d)^*$ define the two adjacent facets F and F' of $\mathcal{P}$. In the special case where no $d+1$ extreme rays lie on a hyperplane, the gift wrapping step corresponds to a simplex pivot (see [16]).

A nice feature of the Adjacency Decomposition Method is that the adjacencies of the most symmetric facets, which are usually the most difficult to treat, may not have to be considered. This is due to the well known *Balinksi's Theoreom* (see e.g. [65, Theorem 3.14]):

Theorem 5.1. [7] *Let $\mathcal{P}$ be a d-dimensional, pointed polyhedral cone. Let G be the undirected graph whose vertices are the facets of $\mathcal{P}$ and whose edges are the ridges of $\mathcal{P}$. Two vertices F_1, F_2 are connected by an edge E if $F_1 \cap F_2 = E$. Then, the graph G is $(d-1)$-connected, i.e. removal of any $d-2$ vertices leaves it connected.*

Using Balinski's Theorem, we know that if the number of facets in unfinished orbits (i.e. those whose neighbors are not known) is less than $d-1$, then any G-inequivalent facet in the unfinished set must be adjacent to some already completely treated facet. This simple criterion has proven [30, 31] to be extremely useful in dealing with examples arising in the geometry of numbers.

5.3. Recursion. Both the Incidence Decomposition Method and the Adjacency Decomposition Method reduce the facet enumeration problem for a d-dimensional polyhedral cone $\mathcal{P}$ to a number of facet enumeration problems for cones in dimension $d-1$. These lower dimensional problems may be too difficult to treat with a standard method as well and therefore we might apply the Incidence or Adjacency Decomposition Method to these lower dimensional problems recursively. So we may speak of the *Recursive Incidence Decomposition Method*, the *Recursive Adjacency Decomposition Method* and the *Recursive Decomposition Method*, if a mixture of both is applied.

The Recursive Incidence Decomposition Method has successfully been used for computing the vertices of the metric polytope MET_8 in [24], whereas the Recursive Adjacency Decomposition Method has successfully been used in [41, 30]. To the best of our knowledge, and to our surprise, a recursive mixture of both has not been used so far.

The crucial steps for the recursion are "$\mathcal{F}_r \leftarrow$ facets of $\mathcal{P}$ incident to r" (where r is some extreme ray) for the Incidence Method (see Section 5.1) and "$\mathcal{F} \leftarrow$ facets of F" (where F is some facet) for the Adjacency Decomposition Method (see Section 5.2). In both cases the problem is to obtain a list of facets for a $(d-1)$-dimensional polyhedral cone F, whose extreme rays are given. Again we

can exploit symmetries. As a result of a call of the Incidence or Adjacency Decomposition Method for F, we obtain a list of G_F-inequivalent facets of F (ridges of $\mathcal{P}$), where G_F is some group acting on the face lattice of F which we have to provide. In a post processing step we then have to obtain a list of $G_\mathcal{P}$-inequivalent (respectively $\mathrm{Stab}(G_\mathcal{P}, F)$-inequivalent) facets of F (ridges of $\mathcal{P}$) out of it.

Note though, that the groups G_F and $G_\mathcal{P}$ may be unrelated, that is, the elements of G_F do not have to be elements of $G_\mathcal{P}$ and vice versa. Think of examples where $\mathcal{P}$ is a polyhedral cone without any symmetries, but with a facet having some symmetries. Vice versa, since we did not assume that G_F is the full (combinatorial) symmetry group, not even the stabilizer $\mathrm{Stab}(G_\mathcal{P}, F)$ of F in $G_\mathcal{P}$ has to be a subgroup of G_F. In the latter case we may simply replace G_F by the group generated by $\mathrm{Stab}(G_\mathcal{P}, F)$ and G_F, so that we assume $\mathrm{Stab}(G_\mathcal{P}, F) \leq G_F$. In this way we possibly enlarge the group and speed up the computations. Moreover, we are able to use the double coset decomposition (see Section 4.3) to split each G_F-orbit $G_F F'$ of facets of F into a finite number of $\mathrm{Stab}(G_\mathcal{P}, F)$-orbits $\mathrm{Stab}(G_\mathcal{P}, F) g_i F'$, where the $g_i \in G_F$ represent the double cosets.

Using the described decomposition methods recursively it might happen that we compute the facets of some sub-cones several times. For example if a face F satisfies $F \lhd F_1 \lhd \mathcal{P}$ for a facet F_1 of a polyhedral cone $\mathcal{P}$, then there is exactly one other facet F_2 such that $F \lhd F_2 \lhd \mathcal{P}$. Hence, if we apply the Adjacency Decomposition Method to F_1 and F_2, then we have to compute the dual description of F two times. Clearly, the number of such repetitions increases as the recursion depth increases. Moreover, equivalent sub-cones may occur in different parts of the face lattice. (Recall that any sub-lattice of the face lattice is the face lattice of some polyhedral cone.) To handle this, we propose to use a *banking system*. That is, given a difficult sub-cone $\mathcal{P}$ of which the facets have been computed up to symmetries, we store with $\mathcal{P}$ (represented by generators) its group of restricted automorphisms and a representative for each facet orbit.

What about the recursion depth needed to practically solve a given polyhedral conversion problem? If we split the problem into subproblems with either decomposition method then some of the subproblems may be easy to solve and may not require a recursive treatment, others might be impossible to treat without such. The implementation [28] allows one to provide a heuristic function to choose whether to recursively apply the Adjacency Decomposition Method, but choosing the right level of recursion requires some trial and error. From our computational experience with this code (for example in [32, 22, 23, 29, 30]), the *incidence number* of a face, that is, the number of extreme rays contained in it, gives a good measure for how difficult the representation conversion is for it. We therefore propose to treat the faces with low incidence numbers first and as much as possible without recursion. For subproblems to be potentially solved with pivoting methods (possibly up to symmetry as below), the "probing" feature of `lrs` can be used to test for expected difficulties of a subproblem.

Finally, let us remark, that parallelization is a possible way to speed up the (recursive) Decomposition Methods as well (see [18, 26]).

6. An incremental method

We briefly sketch here the *Cascade algorithm* by Jaquet [41], which is a symmetry exploiting version of *Fourier–Motzkin elimination*. For this observe first that

a d-dimensional polyhedral cone $\mathcal{P}$ generated by $v_1, \ldots, v_n \in \mathbb{R}^d$ can be obtained as the projection of an n-dimensional polyhedral cone with n linear independent generators $v'_1, \ldots, v'_n$. Without loss of generality, we may assume that $(v_1, \ldots, v_d)$ is a basis of $\mathbb{R}^d$. We set $v'_i = (v_i, 0^{n-d})$ for $i \leq d$ and $v'_i = (v_i, 0^{n-d}) + e_i$ for $i > d$. Let $p_i(x_1, \ldots, x_n) = (x_1, \ldots, x_i, 0, \ldots, 0)$ be the orthogonal projection of $\mathbb{R}^n$ onto $\mathbb{R}^i \times \{0\}^{n-i}$. Then for the cone $\mathcal{P}' = \text{cone}\{v'_1, \ldots, v'_n\}$ we have $\mathcal{P} = p_d(\mathcal{P}')$. The cone $\mathcal{P}'$ is simplicial, since the family $(v'_i)_{1 \leq i \leq n}$ is a basis of $\mathbb{R}^n$.

Take the n-dimensional polyhedral cone $\mathcal{P}'$ and the projection p_{n-1} of $\mathbb{R}^n$ on a $(n-1)$-dimensional hyperplane. The facets of the projection $p_{n-1}(\mathcal{P}')$ are either projections of facets of $\mathcal{P}'$ or projections of intersections of facets of $\mathcal{P}'$. In the Fourier–Motzkin elimination we first compute the facets of $\mathcal{P}'$ and then successively obtain the facets of $p_{n-1}(\mathcal{P}'), \ldots, p_d(\mathcal{P}') = \mathcal{P}$ from one of these two cases. Note, since $\mathcal{P}'$ is a simplex, the facets of $\mathcal{P}'$ are simply given by the $(n-1)$-subsets of extreme rays of $\mathcal{P}'$.

In the Cascade algorithm we also consider symmetries in each step: If G is a group of symmetries of the polyhedral cone $\mathcal{P}$, then the induced symmetry group on $p_i(\mathcal{P}')$ is the stabilizer of the set of vertices $\{v_{d+1}, \ldots, v_i\}$ under G, which we denote by G_i.

So, in order to compute the orbits of facets of $p_i(\mathcal{P}')$ under G_i we need to compute the orbits of facets and ridges of $p_{i-1}(\mathcal{P}')$, first under G_{i-1} and then under G_i using the double coset method.

It is well known that the Fourier–Motzkin method depends in a critical way on the ordering of generators $v_1, \ldots, v_n$ of $\mathcal{P}$ (see [4, 10]). The Cascade Algorithm introduces another such dependency as the size of the set-stabilizers vary enormously according to the chosen ordering. Among the many possible orders of the generators, it is not clear which ordering is best.

In practice Fourier–Motzkin elimination suffers from generating many redundant inequalities. This problem can be eliminated by using the related *Double Description Method* [34]. For a description of how to use the double description method to perform the Fourier–Motzkin projection steps without introducing redundant inequalities, see [9, Section 4.1].

7. A pivoting method

Since for certain classes of input, the most successful methods for the polyhedral representation transformation problem (without taking symmetries into account) are based on the pivot operation of the simplex method, it is natural to consider whether pivoting techniques can be adapted to the symmetric setting.

7.1. The basis graph up to symmetry. For the purposes of this discussion, by *k-basis*, we mean a set of k extreme rays generating the linear span of a k-face of a pointed polyhedral cone. Two k-bases are *adjacent* if they share $k-1$ extreme rays. When k is not specified, we refer to $(d-1)$-bases of a d-dimensional polyhedral cone. Note that the bases of a polyhedral cone depend not only on the combinatorial information contained in the face lattice, but also on the linear dependencies among extreme rays incident to a given face. When discussing $(d-1)$-dimensional polytopes or their d-dimensional homogenizations, we use basis to refer interchangeably to a $(d-1)$-basis of the homogeneous cone or a $(d-2)$-basis of the polytope (i.e., $d-1$ vertices spanning a facet of the polytope).

The *basis graph* has as nodes the bases, and as edges the pairs of adjacent bases. In the so-called *non-degenerate* case each facet of a d-dimensional cone is a $(d-1)$-dimensional simplicial cone. By a perturbation argument, for any cone there exists a triangulation of the boundary which is combinatorially equivalent to the boundary of a polyhedral cone with simplicial facets; it follows by Balinski's Theorem that the basis graph is at least $(d-1)$-connected.

One of the earliest published methods of polyhedral representation transformation [15] is based on an exhaustive exploration of the basis graph. Using the pivot operation of the simplex method, adjacent $(d-1)$-bases can be found in time proportional to the number of input generators and the dimension (for details, see e.g. [3]). Pivoting methods based on Reverse Search [6] have memory usage independent of the output size. In the present work we consider only more direct methods based on generation (and storage) of the basis graph up to symmetry.

In the typical case, generating the entire basis graph is impractical, even if it is not stored. This is because of the enormous number of bases that correspond to each facet in the degenerate (i.e. non-simplicial facet) case. In the non-symmetric setting, perturbation has been widely used to reduce the size of the basis graph under consideration, in particular via the lexicographic (symbolic) perturbation discussed further in Section 7.3.2. We consider one way in which perturbation can be applied in the symmetric case in Section 7.3. There is some tension between the notions of symmetry and perturbation though: whereas perturbation allows us to reduce the size of the basis graph, usually some of the given symmetries are lost. So there is a trade-off in which ideally the quotient graph of the obtained new basis graph with respect to its remaining symmetries is as small as possible.

7.1.1. *Symmetry of the basis graph.* The performance of pivoting based methods under symmetry is determined not by the total size of the basis graph, but by the number of orbits of bases. This number is not determined by the number of orbits of extreme rays and the number of orbits of facets.

We define the *basis automorphism group* to be the subgroup of the combinatorial automorphism group that acts on the basis graph. To see that this is a non-trivial restriction, consider the cone generated by the following (row) vectors in $\mathbb{R}^5$

$$\begin{array}{rrrrr} 1 & 0 & \frac{1}{2} & 1 & 1 \\ -1 & 0 & \frac{1}{2} & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & -1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 0 & -1 & 1 \end{array} \tag{7.1}$$

Combinatorially, it is (the homogenization of) a pyramid over an octahedron, and its combinatorial automorphism group is the 48 element octahedral group. The basis automorphism group has only 16 elements; in particular the orbit of the first d generators (as a basis) is of size 1.

Example 7.1. The regular d-cross polytope C_d has one orbit of d-bases and one orbit of $(d-1)$-bases with respect to its automorphism group.

PROOF. Since the d-cross polytope is simplicial, (i.e., its facets are all simplices) its $(d-1)$-bases are exactly its facets. Observe that at most one pair of vertices in a d-basis must consist of an opposite pair $\pm e_j$, since any two pairs are co-planar. It thus follows from the pigeonhole principle that a d-basis consists of a facet, along with one vertex not on that facet. All such simplices are equivalent under the automorphism group of the cross polytope. □

For a slightly more involved example, let $\mathcal{P} = \mathrm{conv}(v_1, \ldots, v_m) \subset \mathbb{R}^d$ and $\mathcal{Q} = \mathrm{conv}(w_1, \ldots, w_n) \subset \mathbb{R}^e$. The *wreath product* of $\mathcal{P}$ with $\mathcal{Q}$ is defined as

$$\mathcal{P} \wr \mathcal{Q} := \mathrm{conv}\{(\underbrace{0, \ldots, 0}_{d(k-1)}, v_i, \underbrace{0, \ldots, 0}_{d(n-k)}, w_k) : 1 \le i \le m, 1 \le k \le n\} \subset \mathbb{R}^{nd+e}.$$

So loosely speaking, the wreath product of $\mathcal{P}$ and $\mathcal{Q}$ is obtained by attaching to the vertices of $\mathcal{Q}$ pairwise orthogonal subspaces, each containing a copy of $\mathcal{P}$.

Example 7.2. The wreath product $\mathcal{P} \wr \mathcal{Q}$ of a regular d-cross polytope $\mathcal{P}$ with a regular e-cross polytope $\mathcal{Q}$ has dimension $D = 2de + e$, $4de$ vertices, $2^{(d+1)e}$ facets and one orbit of vertices, facets, and $(D-1)$-bases with respect to its linear automorphism group.

PROOF. The proof is based on the results of [42, Section 2.2]. The number of vertices of the cross polytopes $\mathcal{P}$ and $\mathcal{Q}$ are $m = 2d$ and $n = 2e$. The assertion on the dimension and the number of vertices of $\mathcal{P} \wr \mathcal{Q}$ follows immediately from the definition of wreath products. The number of facets and the number of bases orbits can be derived from [42, Proposition 2.2]. By it, the facets of $\mathcal{P} \wr \mathcal{Q}$ are in one-to-one correspondence with choices $(F; F_1, \ldots, F_e)$ of a facet F of $\mathcal{Q}$ and facets F_i of $\mathcal{P}$; which gives the count of facets. Assume w.l.o.g. that F contains the vertices $v_1, \ldots, v_e$ of $\mathcal{Q}$. Then the rows of the following matrix describe the vertices contained in a facet of $\mathcal{P} \wr \mathcal{Q}$:

$$\begin{bmatrix}
\mathcal{P} & & & & & & & & v_1 \\
& \mathcal{P} & & & & & & & v_2 \\
& & \ddots & & & & & & \\
& & & \mathcal{P} & & & & & v_e \\
& & & & F_1 & & & & v_{e+1} \\
& & & & & F_2 & & & v_{e+2} \\
& & & & & & \ddots & & \\
& & & & & & & F_e & v_{2e}
\end{bmatrix}$$

The entries $\mathcal{P}$ and F_i stand for all choices of vertices from $\mathcal{P}$, respectively F_i. So a facet of $\mathcal{P} \wr \mathcal{Q}$ contains $3de$ vertices.

The linear automorphism group of $\mathcal{P} \wr \mathcal{Q}$ contains the *semidirect product* $G = G_P^n \rtimes G_Q$ (where G_P and G_Q denote the automorphism groups of $\mathcal{P}$ and $\mathcal{Q}$ and G_P^n denotes an n-fold direct product). In particular, G_Q is isomorphic to a subgroup of G, which permutes not only $v_1, \ldots, v_{2e}$ in the last e coordinates, but also the n copies of $\mathcal{P}$ attached to them. In contrast to this action interchanging the n d-dimensional subspaces, the elements of the subgroup G_P^n of G act only within these subspaces. By this, all of the vertices and facets lie in one orbit under the action of G. The assertion on the bases follows as well, after noting that a base within a facet (as described above) is determined by e choices of d-bases (in the e copies of $\mathcal{P}$). All of them are equivalent with respect to G by Example 7.1. □

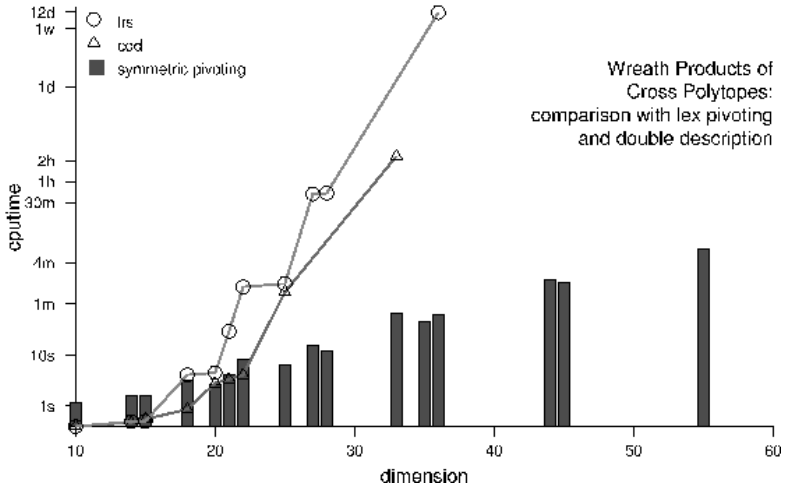

FIGURE 1. Experimental results for wreath products of cross polytopes.

7.2. A prototype implementation. We have implemented a prototype called *symbal* ("Symmetry, Bases, and Lexicography") for facet enumeration up to symmetries via pivoting. The code is mainly in GAP [35], with an external server based on `lrs` doing the pivoting. The main algorithm is a depth first search of the pivot graph; several options are provided for symmetry testing and perturbation (discussed further below). All symmetry groups are represented internally as permutation groups, and the backtracking algorithm of GAP's RepresentativeAction is used to test bases and facets for equivalence. Our main goal with this implementation is not to be able to attack large problems, but to provide a test-bed for better understanding of the main issues involved with pivoting under symmetry. In general we thus are not too concerned with a constant multiplicative slowdown in the runtime of our code versus more polished software; we are more interested in what kinds of problems can be solved in a reasonable amount of time.

At least for certain special cases, our prototype is able to solve quite large problems. Figure 1 compares the run time of `symbal` with `cdd` and `lrs` on the wreath products of cross polytopes discussed in Example 7.2. Of course both of `cdd` and `lrs` compute all of the facets rather than orbits. The times reported here are on a 3GHz Pentium IV with 1G of memory.

We call a cone (or polytope) *basis-simplicial* if it has as few basis orbits as facet orbits. Evidently every simplicial polytope is basis-simplicial. For perhaps less contrived examples of basis-simplicial polytopes than Example 7.2 we mention the Dirichlet–Voronoi-cells (DV-cells) of the root lattices D_3, D_4, and E_8 (convex hull of their shortest non-zero vectors) and some related *universally optimal* spherical polytopes (see [8]). Their boundaries consist of regular cross polytopes and regular simplices only and they thus have at most two orbits of basis with respect to their symmetry group (cf. Example 7.1).

As with the non-symmetric case, for cones with a large number of (orbits of) bases, pivoting is not a good approach, at least not without some kind of perturbation. One benchmark of how *orbitwise degenerate* a polytope is, is whether the number of orbits of bases is larger than the number obtained by triangulating the

TABLE 1. Basis orbits of cubes

Dimension	`lrs` Triangulation	Basis Orbits
4	48	4
5	240	17
6	1440	237
7	10080	9892
8	80640	> 209000

boundary. In this case, one is almost certainly better off applying either `cdd` or `lrs`. A familiar example where this level of degeneracy occurs is in the d-dimensional cubes (see Table 1).

7.3. Orbitwise perturbation. The basis automorphism group could in principle be larger than the linear automorphism group (as a simple example, consider a simplicial polytope combinatorially equivalent to the regular d-cross polytope, but with trivial linear symmetry group). Nonetheless, for the reasons articulated in Section 3, and because linearity simplifies the discussion here, we consider here the case where we are given a subgroup of the restricted isomorphism group of a polyhedral cone.

In this section we consider modifications of the standard *lexicographic perturbation* schemes that preserve some, but not necessarily all of the symmetry of the input. We first characterize one kind of transformation that preserves a prescribed symmetry group.

Proposition 7.3. *Let V be a vector family with restricted automorphism group G. Let $H \leq G$. Let $V_1, V_2, \ldots, V_k$ be the orbits of V under H. Let $u_1, \ldots, u_k$ be a family of fixed points for H. Let*

$$V' = \bigcup_j \{v_i + u_j \mid v_i \in V_j\}$$

Let H' be the restricted automorphism group of V'. Then $H \leq H'$.

PROOF. Let $T \in H$ be a restricted automorphism for V and ρ the corresponding permutation such that $Tv_i = v_{\rho(i)}$. Suppose $v_i \in V_j$. Since there exists an automorphism mapping v_i to $v_{\rho(i)}$, by definition $v_{\rho(i)} \in V_j$. It follows by linearity of T that

$$T(v'_i) = T(v_i + u_j) = Tv_i + Tu_j = v_{\rho(i)} + u_j = v'_{\rho(i)}. \qquad \square$$

Of course, in addition to preserving part of the symmetry group, we need to solve the original representation conversion problem. Let $\nu(\cdot)$ denote the orbitwise perturbation map. The key property we need is that for each facet F of the original cone, there is some basis B such that $\nu(B)$ is a basis of the perturbed cone. The standard way of ensuring this, and the one we adopt here, is to insist that the facets of the perturbed cone induce a subdivision of the facets of the original cone. Several notions of perturbation exist in the literature, including lexicographic [20], numeric [55], symbolic [64], and geometric [56]. We will discuss a method that can be seen as a *linear perturbation* in the language of Seidel [56] or a modified version of the lexicographic perturbation first proposed by Dantzig, Orden and Wolf [20].

Here we take a somewhat weaker definition of perturbation than is typical, since we are not concerned necessarily with obtaining a cone with simplicial facets as result. Let V be vector family. We say $W \subseteq V$ is *extreme* (for V) if it is contained in some facet of cone(V).

Definition 7.4. We say that $\widetilde{V}$ is a *valid perturbation* of V if there is a bijection $\nu(\cdot)$ between V and $\widetilde{V}$ such that for any $W \subseteq V$,

(1) If $\nu(W)$ is linearly dependent then W is.
(2) If $\nu(W)$ is extreme for $\widetilde{V}$ then W is extreme for V.

From Definition 7.4, we have immediately the following refinement property.

Proposition 7.5. *For a valid perturbation $\widetilde{V}$ of a vector family V we have*

(1) *The boundary complex of* cone$(\widetilde{V})$ *is a subdivision of the boundary complex of* cone(V).
(2) *If $\widetilde{X}$ is a valid perturbation of $\widetilde{V}$, then $\widetilde{X}$ is a valid perturbation of V.*

The following proposition shows that valid perturbations result from sufficiently small changes to a vector family.

Proposition 7.6. *For any vector family $V \subset \mathbb{R}^d$, for any $W \subseteq V$ and any vector $u \in \mathbb{R}^d$, $\widetilde{V}(\varepsilon) = \{w + \varepsilon u : w \in W\} \cup V \setminus W$ is a valid perturbation of V for all ε with $|\varepsilon|$ sufficiently small.*

Proof. Consider the vector family $\widetilde{V}(\varepsilon)$ changing as ε varies. From the point of view of linear dependence, we need only concern ourselves with a d-set of vectors $v_1, \ldots, v_d$, which are independent in V, but whose perturbation becomes dependent for $\varepsilon > 0$. The following continuity argument shows that for small enough ε no new linear dependencies may occur. Let T be the matrix whose rows are $v_1, \ldots, v_d$. By renumbering we may assume the first k rows of T are in W. Let U be the matrix whose first k rows are u and the remaining rows are the zero vector. Then

$$g(\varepsilon) = \det(T + \varepsilon U)$$

is a polynomial in ε, which is non-zero in a neighborhood of $\varepsilon = 0$. Taking the intersection of all such intervals over all full rank d-sets of vectors yields (1) and (2) of Definition 7.4 □

Let V be a vector family with restricted automorphism group G. Let H be a subgroup of G. Let u_H be a fixed point for H. We say that $\widetilde{V}$ is obtained by *pushing* $W \subseteq V$ (respectively *pulling* $W \subseteq V$) if

$$\widetilde{V} = \{w + \sigma\varepsilon u_H \mid w \in W\} \cup V \setminus W$$

where $\sigma = 1$ (respectively $\sigma = -1$) and $\varepsilon > 0$ is sufficiently small so that $\widetilde{V}$ is a valid perturbation of V.

Let $V_1, V_2, \ldots V_k$ be the orbits of V with respect to H. In general the combinatorial structure of the resulting boundary complex depends not only whether each V_j is pulled or pushed, but on the order these operations are carried out. We say that $\widetilde{V}$ is an *orbitwise lexicographic perturbation* of V with respect to H if it is obtained by pulling or pushing each orbit defined by H in some *perturbation order* π. Considering the special case where V contains the homogenization of the vertices of a d-polytope and u_H is the homogenization of the origin, pulling (respectively

pushing) corresponds to scaling an orbit outward (respectively inward) with respect to the origin; this terminology is consistent with that of Lee [47].

Combining Propositions 7.3, 7.5, and 7.6, we have:

Proposition 7.7. *If $\widetilde{V}$ is an orbitwise lexicographic perturbation of V with respect to a subgroup H of V's restricted automorphism group, then H acts on the basis graph of* $\mathrm{cone}(\widetilde{V})$.

In the case that there is only one orbit under the basis automorphism group, perturbing all input vectors by the same vector will not decrease the degeneracy of the problem. Our general strategy will thus be to choose a subgroup H that has multiple orbits of extreme rays, and at the same time choose different perturbations u_i for each orbit of V with respect to H.

It is known [47] that lexicographic perturbation of all of the vertices induces a triangulation (also called lexicographic) of the boundary of the polytope. In our case because we perturb (i.e. push or pull) all vertices in an orbit by the same amount, we cannot in general guarantee a simplicial result (i.e. an induced triangulation). Thus we continue to explore the complete basis graph of the perturbed cone (polytope), up to equivalence classes of bases. We nonetheless hope for a significant reduction in the size of the basis graph by (effectively) breaking up very degenerate facets of the cone. The trade-off, examined further in the next section, is that although we may lose some symmetry of the polyhedron, the quotient graph of the new basis graph (with respect to the remaining symmetries) may become smaller. Moreover, the degree of the vertices in the basis graph may have decreased, which speeds up the computation of the quotient.

7.3.1. *Choosing a subgroup to preserve.* For any polyhedral cone $\mathcal{P}$ with symmetry group G, in order to effectively use orbitwise perturbation, one needs to find a subgroup $H \leq G$ such that the above mentioned possible computational gain is as large as possible. In order to develop some heuristics for how one might find such a subgroup, we have systematically studied the subgroups of the restricted automorphism group of the DV-cell of the E_7 root lattice (hereafter we use $\mathrm{conv}\,\mathsf{E}_7$ to denote this DV-cell). Although we do not claim the results from a single example are in any way conclusive, they do at least suggest some ideas for further study.

Let G denote the restricted automorphism group of $\mathrm{conv}\,\mathsf{E}_7$. This is a group of order 2903040 that has one orbit on the 126 vertices of E_7. The polytope $\mathrm{conv}\,\mathsf{E}_7$ has 632 facets in 2 orbits. It has 161 basis orbits, compared to 20520 bases in the triangulation produced by `lrs`.

The experiments on this section were carried out an Acenet (`http://www.ace-net.ca`) cluster with SunFire x4100 nodes (two 2.6 GHz dual-core Opteron 285 SE processors and 4 GB RAM per core).

A sample of conjugacy classes of the subgroup lattice was generated using the GAP function `LatticeByCyclicExtension`, with the restriction that groups of size less than or equal to 100 were discarded. A representative was chosen from each conjugacy class, yielding 102 subgroups of G. In general combinatorially distinct perturbations could result from choosing a different perturbation order, and by varying the direction (push or pull) that each orbit is perturbed. In this experiment we restricted ourselves to orbitwise pulling, and ordered according to the smallest index in an orbit. Orbitwise pulling perturbations were generated for each of these subgroups, and these perturbations were used as input to our prototype `symbal`. A total of 99 of these computations completed in the time allocated. The fastest time

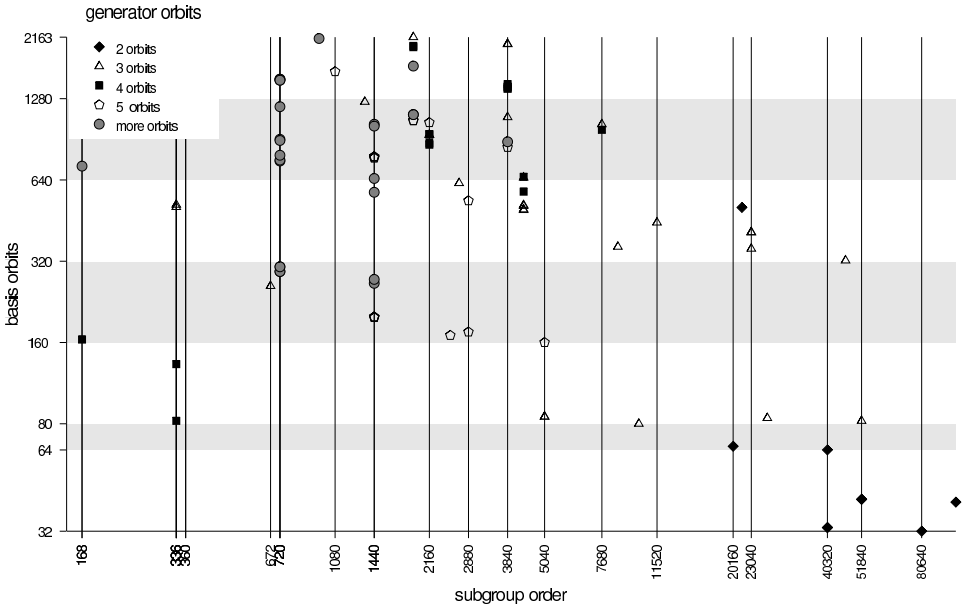

FIGURE 2. Number of basis orbits for various subgroups on conv E_7.

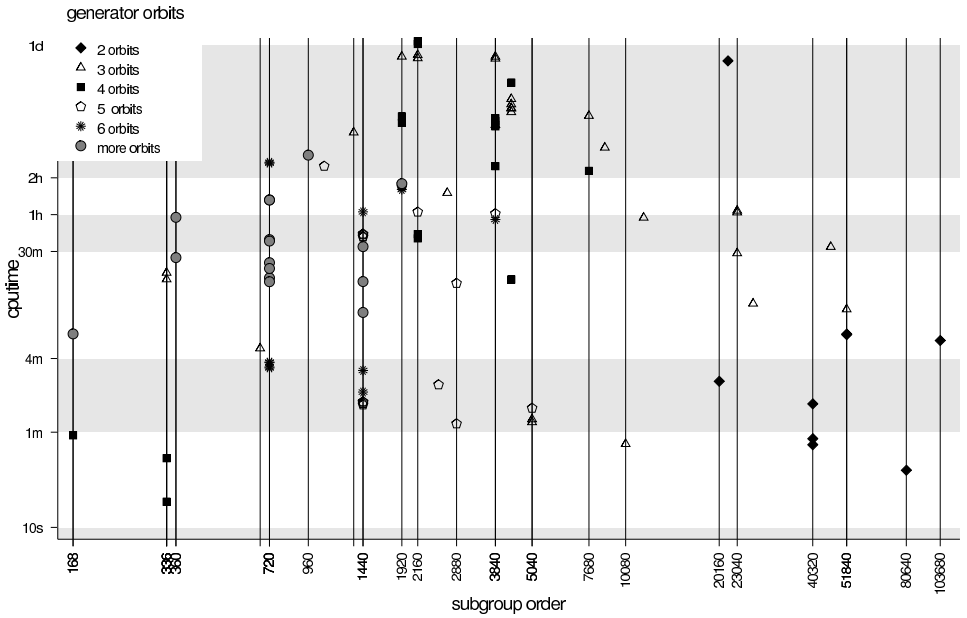

FIGURE 3. CPU time to enumerate facets for various subgroups on conv E_7.

was about 16 seconds, and longest just over 26 hours. The unperturbed version completed in about half an hour. Perturbation thus yielded results ranging from more than a 100-fold speedup to more than 50-fold slow down.

Figure 2 presents the experimental results in terms of the number of basis orbits computed. The groups are classified according to their order (x-axis) and number of orbits of vertices (shown by symbol). Recall that for any actual perturbation, we need at least two generator orbits. Of the subgroups with less than 64 orbits of (perturbed) bases all had two generator (i.e. vertex) orbits. The best result is 32 basis orbits, achieved by a subgroup of order 80640. Within the class of subgroups having two generator orbits, the best results were achieved by groups of large order. Note that a large subgroup inducing two (or any other fixed small number) of generator orbits can be found by choosing random sets of elements as generators.

Comparing Figure 2 with Figure 3, we see runtime is not completely a function of the number of basis orbits, but the size of the group also plays a role. The actual best time is achieved by a group of order 336, with 4 generator orbits and 82 basis orbits. After a certain point a small subgroup is no longer advantageous, since the number of basis orbits are too large. In the limiting case, with a trivial group, our pivoting scheme produces a lexicographic triangulation like `lrs`; albeit less efficiently in our prototype implementation. A relatively small subgroup with a fixed number of input orbits can also be found by the same random sampling strategy.

Figure 4 illustrates our two heuristics applied to the three dimensional cube. On the left we have the triangulation induced by a subgroup of order 24 having two generator orbits, In the center, the triangulation is induced by the stabilizer of a pair of opposite vertices. On the right, we take the same stabilizer, but push the first orbit instead of pulling.

The triangulation induced by pulling opposite vertices of a 3-cube turns out to be a special case of a triangulation of the d-cube that has only one basis orbit in any dimension. Let $I^d = [-1,1]^d$ denote the centrally symmetric d-cube. Define $\mathbf{e} = \sum_{i=1}^{d} e_i$ For each permutation $\rho \in \mathrm{Sym}(d)$, let Δ_ρ denote the simplex with vertices:

$$-\mathbf{e}, \quad -\mathbf{e} + 2(e_{\rho(1)}), \quad -\mathbf{e} + 2(e_{\rho(1)} + e_{\rho(2)}), \ldots, \quad \mathbf{e}.$$

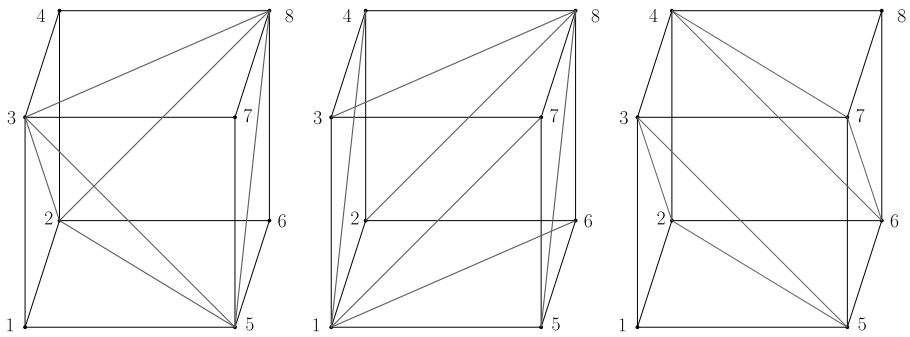

FIGURE 4. Symmetric triangulations of the boundary of the 3-cube; in each case one orbit push or pull suffices to triangulate.

Let Δ denote the union of all Δ_ρ and let $\bar{\Delta}$ denote the set of $(d-1)$-simplices formed by intersecting the simplices of Δ with the boundary of I^d. It is known [21] that Δ forms a triangulation of I^d; consequently $\bar{\Delta}$ forms a triangulation of the boundary, since every triangulation of a polytope induces a triangulation of its boundary. We call Δ (respectively $\bar{\Delta}$) the *linear ordering triangulation* of I^d (respectively of the boundary of I^d.) Since there is a bijection between permutations in $\mathrm{Sym}(d)$ and simplices in Δ, $\mathrm{Sym}(d)$ acts transitively on Δ by permuting coordinates.

Example 7.8. Let V denote the vertices of I^d, i.e. $V = \{\pm 1\}^d$. Let H denote the stabilizer of the automorphism group of I^d on $\{-\mathbf{e}, \mathbf{e}\}$. Define $\omega(v) = \min(\mathbf{e}^T v, -\mathbf{e}^T v)$. Let $\nu(V)$ denote the H-orbitwise pulling of V in the order induced by ω. Then the following holds:

(a) H acts transitively on $\bar{\Delta}$.
(b) $\partial \operatorname{conv} \nu(V)$ is combinatorially equivalent to $\bar{\Delta}$.

Proof. Let us first remark that the order of orbits induced by ω is well defined. The group $H \leq \mathrm{Sym}(2^d)$ is generated by a set of generators permuting coordinates, along with a *switching permutation* σ that maps v to $-v$. It follows that the H-orbits of V are the equivalence classes of V with respect to ω.

To see (a), consider the d-simplex (corresponding to the identity permutation) Δ_{Id} with vertices $\{-\mathbf{e}, -\mathbf{e}+2e_1, -\mathbf{e}+2(e_1+e_2), \ldots, \mathbf{e}\}$. Let σ' denote the permutation in H that first applies σ (i.e. switching) followed by reversing the order of coordinates. The permutation σ' is an automorphism of Δ_{Id} which carries the $(d-1)$-simplex $\{-\mathbf{e}, -\mathbf{e}+2e_1, \ldots, e-2e_d)\}$ to $\{\mathbf{e}, \mathbf{e}-2e_d, \ldots, -\mathbf{e}+2e_1\}$. These two $(d-1)$ simplices are precisely the contribution of Δ_{Id} to the linear ordering triangulation $\bar{\Delta}$. Thus any simplex of $\bar{\Delta}$ can be mapped to any other by an action of $\mathrm{Sym}(d)$ on Δ (i.e. permuting coordinates), followed by possibly applying σ'.

We now consider (b). We argue that $\nu(V)$ induces a linear ordering triangulation of each k-face of I^d, $0 \leq k < d$. Each $(d-2)$-face will receive the same (linear ordering) triangulation from the two facets that contain it, hence the triangulations of the facets form a triangulation of the boundary.

For $0 \leq k \leq 2$, there is nothing to prove. Let F be a k-face of I^d, $2 < k < d$. Let v^+ (resp. v^-) be the vertex of F with the most positive (resp. negative) coordinates. Recall that we will first pull the H-orbit with smallest ω value.

If the functional ω is minimized uniquely at $v^* \in \{v^-, v^+\}$ then the perturbation corresponds locally to a standard [47] pulling and the corresponding subdivision is into pyramids $F_1, \ldots, F_j$ with apex v^* and bases corresponding to all of the $k-1$ faces of F that do not contain v^*.

Otherwise ω is minimized at both v^- and v^+. The perturbation thus takes v^+ to ρv^+ and v^- to some ρv^-, $\rho > 1$. This turns out to induce a subdivision of F into polytopes F_j^2 with vertices $V_j = \{\rho v^-, \rho v^+\} \cup R_j$ where R_j is the vertex set of a $(k-2)$-face of F containing neither v^- nor v^+. The polytope F_j^2 is a *2-fold pyramid*, since $v^- \notin \operatorname{aff} R_j$ and $v^+ \notin \operatorname{aff}(\{v^-\} \cup R_j)$. It follows that $\dim F_j^2 = \dim F = k$.

That the F_j^2 are induced by the perturbation can be seen by exhibiting a supporting hyperplane of $\operatorname{conv} \nu(V)$. Without loss of generality, let F be defined by equations $x_i = 1$, $i = 1, \ldots, d-k$. The $(k-2)$-face $\operatorname{conv}(R_j)$ must be defined by further equations $x_p = 1$, $x_q = -1$, $p, q > d-k$. Let $\mu = 1/\rho$. Consider the hyperplane $h_j = \{x \mid a^T x = 1\}$, where

$$a_i = \begin{cases} \mu/(d-k) & 1 \le i \le d-k, \\ (1-\mu)/2 & i = p, \\ -(1-\mu)/2 & i = q, \\ 0 & \text{otherwise.} \end{cases}$$

It can be verified that h_j supports $\operatorname{conv}\nu(V)$ and $h_j \cap \operatorname{conv}\nu(V) = F_j^2$.

It remains to see that the F_j^2 cover F, i.e. that there are no other cells in the induced subdivision. Consider an arbitrary relative interior point x of F. Let r be the ray from ρv^- through x. Let y be the first intersection of r with ∂F after x. For each $(k-1)$-face of F, the double pulling of $\{ v^+, v^- \}$ acts like a single pulling decomposing the boundary of F into pyramids with apex either ρv^- or ρv^+; hence $y \in \partial F_j^2$ for some j. It follows that x is in F_j^2.

Now suppose for all $j < k$, $\nu(V)$ induces a linear ordering triangulation of the j-faces of I^d. From the refinement property Proposition 7.5, we know that $\nu(V)$ induces a decomposition of the pyramids $F_1, \ldots, F_k$ (respectively of the 2-fold pyramids $F_1^2, \ldots, F_{k(k-1)}^2$). In both cases the resulting k-simplices correspond to the coordinatewise-monotone paths from v^- to v^+ in F. □

7.3.2. *Symbolic implementation.* Suppose we are given $V' \subset \mathbb{R}^{d+1}$ which are homogeneous coordinates for the generators V of some polyhedron $\mathcal{P} \subset \mathbb{R}^d$. We consider V here as a matrix (with v_i as rows), and let u denote the column vector of corresponding $(d+1)$st coordinates in V'. Following the conventions of Section 2, we suppose u_j is 1 if v_j is a vertex, and 0 if it is an extreme ray. We consider the polyhedron $\mathcal{P}^\diamond$

$$\mathcal{P}^\diamond = \{x \in \mathbb{R}^d : Vx \ge -u\}. \tag{7.2}$$

The polyhedron $\mathcal{P}^\diamond$ may be thought of as $\operatorname{cone}(V')^* \subset (\mathbb{R}^{d+1})^*$ intersected with the hyperplane $x_{d+1} = 1$. By duality, to find the generators of $\mathcal{P}^\diamond$ is equivalent to finding the facets of $\mathcal{P}$.

Let G be the restricted automorphism group of V. Let G' be the induced group acting on V'. We will assume without loss of generality that the origin is the centroid of V and thus a fixed point of G. It follows that e_{d+1} is a fixed point of G'. Applying Proposition 7.3 to our dual representation (7.2), we see that orbitwise perturbing the right-hand side vector according $H \le G$ will preserve the symmetries of H. The perturbed system thus has the form

$$Vx \ge -(u + \mu), \tag{7.3}$$

where $\mu_i = \sigma_j \varepsilon_j$ for $\sigma_j \in \{\pm 1\}$ and j is the index of the orbit containing v_i. To ensure an orbitwise lexicographic perturbation, we will insist

$$1 \gg \varepsilon_1 \gg \varepsilon_2 \gg \cdots \gg \varepsilon_k > 0, \tag{7.4}$$

where by $x \gg y$ we mean that y is much smaller than x, i.e. it is not possible to combinatorially change the polyhedron defined by (7.3) by choosing $y > 0$ smaller. To implement this symbolically, we need a modification of the standard lexicographic pivot rule (see [3] or [16] for more details). Let $b = -(u + \mu)$ and $A = [V - I]$. After adding *slack variables* to (7.3), we are left with a system of the form

$$Ax = b$$

with n rows and $n+d$ columns, where the first d columns are the *decision variables*.

A *feasible basis* consists of a partition (β, η) of the column indices such that A_β (columns of A indexed by β) is non-singular and any slack variables in $x^*_\beta = A_\beta^{-1} b$ are non-negative. In order to move from one feasible basis to another, we need to perform a *pivot*. We start by choosing a column index j to leave β. To find a column index to replace j, we need to find

$$\operatorname{argmin}_i \frac{b^*_i}{a_i}$$

where $b^* = A_\beta^{-1} b$ and $a = A_\beta^{-1} A_\eta$. In our case, where $b = -u - \mu$, we may decompose b^*, and thus the ratio test into two parts $A_\beta^{-1}(-u)$ and $A_\beta^{-1}(-\mu)$. Because the values of the ε_j are chosen very small, the second part is considered only to break ties. We write

$$A_\beta^{-1}(-\mu) = N \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_k \end{bmatrix}$$

where column j of N is defined by summing the columns of A_β^{-1} corresponding to orbit j of generators, and multiplying by $-\sigma_j$. Because of the ordering (7.4), in order to evaluate

$$\operatorname{argmin}_i \frac{(N\varepsilon)_i}{a_i}$$

we proceed column by column in N, reducing the set of ties at each iteration.

7.4. Other refinements and implementation details.

7.4.1. *Adjacency decomposition pruning.* Consider facets F_0 and F_1 that are equivalent under some symmetry of the basis automorphism group. This same symmetry acts as an isomorphism between the corresponding basis graphs. It follows that when we discover a basis B defining a new orbit, but the facet F spanned by B is known, we do not need to explore the neighbours of B since they will be explored in our canonical (i.e. discovered first) facet in the orbit of F. In order to ensure that orbits are not discarded, we are careful not to mark B as known until its canonical discovery.

Although this pruning does not reduce the number of orbits of bases explored, it can reduce the number of actual bases visited (and tested for isomorphism), since bases of a given orbit are not revisited in every copy of the facet F. As an example, consider the 3 quadrilateral facets illustrated in Figure 5, with rotational symmetry yielding 4 orbits of bases.

Without pruning, a depth first search visits all of the bases; with pruning only 8 of 12 bases are visited (see Figure 6). The speedup obtained depends roughly on the number of facets visited by the unpruned search, which is bounded by the number of basis orbits.

7.4.2. *Metric invariants.* In the case where our symmetry group preserves the inner product between pairs of vectors, as is the case for the restricted automorphisms discussed in this paper, we may take advantage of this in several ways.

For any face or basis X to be tested for isomorphism, we may construct a graph (analogous to that constructed in Proposition 3.1) whose nodes are the vectors of X and whose edges are the angles between them. This graph contains geometric information not present in the index sets representing X, which can help to speed

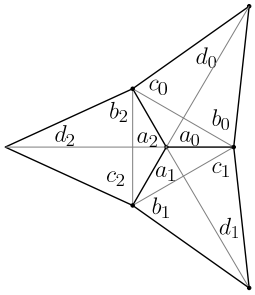

FIGURE 5. 3 facets with a 3-fold rotational symmetry. Basis q_i has vertices p_i, q_i, r_i around the boundary of the facet.

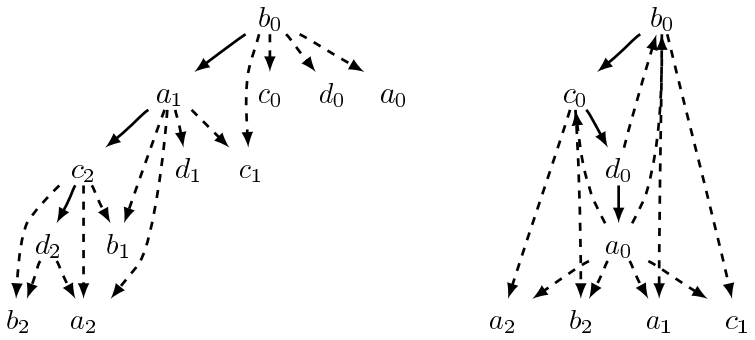

FIGURE 6. Comparing the search with pruning (right) and without (left).

up an algorithm to find an isomorphism. A simpler observation, and equally widely applicable, is that the set of pairwise inner products of two isomorphic faces or bases must be equal. This allows us to store orbit representatives in a data structure such as a hash table or a balanced tree, with the key to the data structure being the set of inner products. This permits more efficient isometry testing by retrieving exactly those orbit representatives which pass the inner product invariant.

It is computationally easy to test whether a given linear transformation T is in the restricted automorphism group $\mathrm{Aut}(V)$ of vector family V. Since we are interested in restricted automorphisms carrying basis X to basis Y, we can additionally test if $\Pi XT = Y$ for some $\Pi \in \mathrm{Aut}(X)$ (where $\mathrm{Aut}(X)$ can be computed by the same techniques as Proposition 3.1). We have only implemented an exhaustive search of $\mathrm{Aut}(X)$, and this is naturally only effective when $\mathrm{Aut}(X)$ is quite small. In principle it should be possible to integrate the test for T being a restricted isomorphism into a backtracking procedure to search for Π.

8. Conclusions

Much as in the case of polyhedral representation conversion without symmetries, a certain amount of trial and error seems to be necessary to decide on the best method to attack a given conversion problem up to symmetries. Currently decomposition methods have the best record of solving interesting problems; on the other hand current software requires a certain amount of user intervention in the

form of choosing how to treat subproblems. It would be helpful to automate this process. In this context, a virtue of the pivoting methods is that good methods to estimate their running time exist [5]. It would be beneficial, not just when working with symmetry, to have effective methods (or at least heuristics) for estimating the running time of incremental methods.

References

1. M. M. Anzin, *On the density of a lattice covering for $n = 11$ and $n = 14$*, Tr. Mat. Inst. Steklova **239** (2002), Diskret. Geom. i Geom. Chisel, 20–51 (Russian); English transl., Proc. Steklov Inst. Math. **239** (2002), no. 4, 13–44.
2. D. Avis, *lrs home page*, School of Computer Science, McGill University, 1993; `http://www-cgrl.cs.mcgill.ca/~avis/C/lrs.html`.
3. ———, *A revised implementation of the reverse search vertex enumeration algorithm*, Polytopes—Combinatorics and Computation (Oberwolfach, 1997), DMV Sem., vol. 29, Birkhäuser, Basel, 2000, pp. 177–198.
4. D. Avis, D. Bremner, and R. Seidel, *How good are convex hull algorithms*?, Comput. Geom. **7** (1997), 265–301.
5. D. Avis and L. Devroye, *Estimating the number of vertices of a polyhedron*, Snapshots in Computational and Discrete Geometry **3** (1994), 179–190.
6. D. Avis and K. Fukuda, *A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra*, Discrete Comput. Geom. **8** (1992), no. 3, 295–313.
7. M. L. Balinski, *On the graph structure of convex polyhedra in n-space*, Pacific J. Math. **11** (1961), 431–434.
8. B. Ballinger, G. Blekherman, H. Cohn, N. Giansiracusa, E. Kelly, and A. Schürmann, *Experimental study of energy-minimizing point configurations on spheres*, Exp. Math. (to appear).
9. D. Bremner, *On the complexity of vertex and facet enumeration for convex polytopes*, Ph.D. Thesis, School of Computer Science, McGill University, 1997; `http://www.cs.unb.ca/profs/bremner/papers/phd`.
10. D. Bremner, *Incremental convex hull algorithms are not output sensitive*, Discrete Comput. Geom. **21** (1999), no. 1, 57–68.
11. D. Bremner, K. Fukuda, and A. Marzetta, *Primal-dual methods for vertex and facet enumeration*, Discrete Comput. Geom. **20** (1998), no. 3, 333–357.
12. G. Brinkmann, *Isomorphism rejection in structure generation programs*, Discrete Mathematical Chemistry (New Brunswick, 1998), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 51, Amer. Math. Soc., Providence, RI, 2000, pp. 25–38.
13. C. S. Chan and D. P. Robbins, *On the volume of the polytope of doubly stochastic matrices*, Experiment. Math. **8** (1999), no. 3, 291–300.
14. D. R. Chand and S. S. Kapur, *An algorithm for convex polytopes*, J. Assoc. Comput. Mach. **17** (1970), 78–86.
15. A. Charnes, *The simplex method: optimal set and degeneracy*, An Introduction to Linear Programming, Lecture VI, John Wiley & Sons, Inc., New York; Chapman & Hall, Ltd., London, 1953, pp. 62–70.
16. V. Chvátal, *Linear programming*, Ser. Books Math. Sci., W. H. Freeman and Company, New York, 1983.
17. T. Christof and A. Löbel, PORTA: *polyhedron representation transformation algorithm* (v. 1.3.1), 1997; `http://www.zib.de/Optimization/Software/Porta/`.
18. T. Christof and G. Reinelt, *Decomposition and parallelization techniques for enumerating the facets of combinatorial polytopes*, Internat. J. Comput. Geom. Appl. **11** (2001), no. 4, 423–437.
19. ———, *Combinatorial optimization and small polytopes*, Top **4** (1996), no. 1, 1–64.
20. G. B. Dantzig, A. Orden, and P. Wolfe, *The generalized simplex method for minimizing a linear form under linear inequality restraints*, Pacific J. Math. **5** (1955), 183–195.
21. J. De Loera, J. Rambau, and F. Santos, *Triangulations of point configurations*, Springer (to appear).
22. M. Deza and M. Dutour, *Cones of metrics, hemi-metrics and super-metrics*, Annals of the European Academy of Sciences (2003), 141–162.

23. M. Deza, M. Dutour, and E. Panteleeva, *Small cones of oriented semi-metrics*, Amer. J. Math. Management Sci. **22** (2002), no. 3-4, 199–225.
24. A. Deza, K. Fukuda, T. Mizutani, and C. Vo, *On the face lattice of the metric polytope*, Discrete and Computational Geometry, Lecture Notes in Comput. Sci., vol. 2866, Springer, Berlin, 2003, pp. 118–128.
25. A. Deza, K. Fukuda, D. Pasechnik, and M. Sato, *On the skeleton of the metric polytope*, Discrete and Computational Geometry (Tokyo, 2000), Lecture Notes in Comput. Sci., vol. 2098, Springer, Berlin, 2001, pp. 125–136.
26. A. Deza and G. Indik, MPI-OWE*, parallel orbitwise vertex enumeration algorithm*, 2006; `http://www.cas.mcmaster.ca/~deza/mpiowe.html`.
27. ———, *A counterexample to the dominating set conjecture*, Optim. Lett. **1** (2007), no. 2, 163–169.
28. M. Dutour Sikirić, *Polyhedral home page*, `http://www.liga.ens.fr/~dutour/Polyhedral`.
29. M. Dutour Sikirić, A. Schürmann, and F. Vallentin, *A generalization of Voronoi's reduction theory and applications*, Duke Math. J. **142** (2008), no. 1, 127–164.
30. ———, *Classification of eight dimensional perfect forms*, Electron. Res. Announc. Amer. Math. Soc. **13** (2007), 21–32.
31. ———, *Complexity and algorithms for computing Voronoi cells of lattices*, (submitted); available at `arXiv:0804.0036`.
32. M. Dutour and F. Vallentin, *Some six-dimensional rigid lattices*, Voronoi's Impact on Modern Science, Book 3 (H. Syta, A. Yurachivsky, P. Engel, eds.), Proc. Inst. Math. Nat. Acad. Sci. Ukraine, vol. 55, 2005, pp. 102–108.
33. K. Fukuda, *`cdd+` reference manual*, Inst. Oper. Res., Swiss Federal Inst. of Technology, Zurich, Switzerland, 1995; `http://www.ifor.math.ethz.ch/~fukuda/cdd_home/cdd.html`.
34. K. Fukuda and A. Prodon, *Double description method revisited*, Combinatorics and Computer Science (Brest, 1995), Lecture Notes in Comput. Sci., vol. 1120, Springer, Berlin, 1996, pp. 91–111.
35. *The* GAP *group* (v. 4.4.6), GAP—Groups, Algorithms, and Programming, 2005; `http://www.gap-system.org`.
36. E. Gawrilow and M. Joswig, *Polymake*: *a framework for analyzing convex polytopes*, Polytopes—Combinatorics and Computation (Oberwolfach, 1997), DMV Sem., vol. 29, Birkhäuser, Basel, 2000, pp. 43–73.
37. V. P. Grishukhin, *Computing extreme rays of the metric cone for seven points*, European J. Combin. **13** (1992), 153–165.
38. B. Grünbaum, *Convex polytopes*, 2nd ed., Springer, New York, 2003.
39. T. Grüner, R. Laue, and M. Meringer, *Algorithms for group actions applied to graph generation*, Groups and Computation II (New Brunswick, 1995), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 28, Amer. Math. Soc., Providence, RI, 1997, pp. 113–122.
40. D. F. Holt, B. Eick, and E. A. O'Brien, *Handbook of computational group theory*, Discrete Math. Appl. (Boca Raton), Chapman & Hall/CRC, Boca Raton, FL, 2005.
41. D. O. Jaquet, *Énumération complète des formes parfaites en dimension* 7, Ann. Inst. Fourier (Grenoble) **43** (1993), no. , 21–55.
42. M. Joswig and F. H. Lutz. *One-point suspensions and wreath products of polytopes and spheres*, J. Combin. Theory Ser. A **110** (2005), no. 2, 193–216.
43. M. Joswig and G. M. Ziegler, *Convex hulls, oracles, and homology*, J. Symbolic Comput. **38** (2004), no. 4, 1247–1259.
44. V. Kaibel and A. Schwartz, *On the complexity of polytope isomorphism problems*, Graphs Combin. **19** (2003), no. 2, 215–230.
45. P. Kaski and P.R.J. Östergård, *Classification algorithms for codes and designs*, Algorithms Comput. Math., vol. 15, Springer-Verlag, Berlin, 2006.
46. A. Kerber, *Applied finite group actions*, 2nd ed., Algorithms Combin., vol. 19, Springer-Verlag, Berlin, 1999.
47. C. W. Lee, *Regular triangulations of convex polytopes*, Applied Geometry and Discrete Mathematics, DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 4, Amer. Math. Soc., Providence, RI, 1991, pp. 443–456.
48. J. S. Leon, *Permutation group algorithms based on partitions,* I. *Theory and algorithms*, J. Symbolic Comput. **12** (1991), no. 4-5, 533–583.

49. ———, *Partitions, refinements, and permutation group computation*, Groups and Computation, II (New Brunswick, 1995), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 28, Amer. Math. Soc., Providence, RI, 1997.997, pp. 123–158.
50. E. M. Luks, *Permutation groups and polynomial-time computation*, Groups and Computation (New Brunswick, 1991), DIMACS Ser. Discrete Math. Theoret. Comput. Sci., vol. 11, Amer. Math. Soc., Providence, RI, 1993, pp. 139–175.
51. T. H. Matheiss and D. S.Rubin, *A survey and comparison of methods for finding all vertices of convex polyhedral sets*, Math. Oper. Res. **5** (1980), no. 2, 167–185.
52. A. Marzetta, pd—C-*implementation of the primal dual algorithm*, 1997; `http://www.cs.unb.ca/profs/bremner/pd/`.
53. B. D. McKay, `nauty`, 2005; `http://cs.anu.edu.au/people/bdm/nauty/`.
54. ———, nauty *user's guide* (version 2.4), 2006; `http://cs.anu.edu.au/people/bdm/nauty/nug-2.4b3.pdf`.
55. N. Megiddo and R. Chandrasekaran, *On the ε-perturbation method for avoiding degeneracy*, Oper. Res. Lett. **8** (1989), 305–308.
56. R. Seidel, *The nature and meaning of perturbations in geometric computing*, Discrete Comput. Geom. **19-1** (1998), 1–17.
57. A. Schrijver, *Theory of linear and integer programming*, Wiley-Intersci. Ser. Discrete Math., A Wiley-Interscience Publication, John Wiley & Sons, Ltd., Chichester, 1986.
58. ———, *Combinatorial optimization. Polyhedra and efficiency.* C. *Disjoint paths, hypergraphs*, Algorithms Combin., vol. 24 C, chapters 70–83, Springer-Verlag, Berlin, 2003.
59. A. Seress, *Permutation group algorithms*, Cambridge Tracts in Math., vol. 152. Cambridge University Press, Cambridge, 2003.
60. B. Sturmfels and J. Yu, *Classification of six-point metrics*, Electron. J. Combin. **11** (2004); Research Paper R44.
61. G. Swart, *Finding the convex hull facet by facet*, J. Algorithms **6** (1985), 17–48.
62. G. F. Voronoi, *Nouvelles applications des paramètres continues à la théorie des formes quadratiques*, Sur Quelques Propriétés des Formes Quadratiques Positives Parfaites (1er Mémoire), J. Reine Angew. Math. **133** (1908), 97–178.
63. ———, *Nouvelles applications des paramètres continus à la théorie des formes quadratiques*, Recherches sur les Parallélloedres Primitifs (2e Mémoire), J. Reine Angew. Math. **134** (1909), 198–287; ibid. **136**, 67–181.
64. C.-K. Yap, *Symbolic treatment of geometric dependancies*, J. Symbolic Comput. **10** (1990), no. 3-4, 349–370.
65. G. Ziegler, *Lectures on polytopes*, Grad. Texts in Math., vol. 152, Springer-Verlag, New York, 1995.

Faculty of Computer Science and Department of Mathematics ans Statistics, University of New Brunswick, Box 4400, Fredericton, NB E3B 5A3, Canada
E-mail address: `bremner@unb.ca`

Group for Satellite Oceanography, Department of Marine and Environmental Research, Rudjer Bosković Institute, Bijenicka 54, 10000 Zagreb, Croatia
E-mail address: `mdsikir@irb.hr`

Mathematics Department, University of Magdeburg, 39106 Magdeburg, Germany
E-mail address: `achill@math.uni-magdeburg.de`