On the Hardness of Computing Intersection, Union and Minkowski Sum of Polytopes

Hans Raj Tiwary

hansraj@cs.uni-sb.de

FR Informatik

Universität des Saarlandes

D-66123 Saarbrücken, Germany

Tel: +49 681 3023235

Fax: +49 681 3025576

Abstract

For polytopes $P_1, P_2 \subset \mathbb{R}^d$ we consider the intersection $P_1 \cap P_2$, the convex hull of the union $CH(P_1 \cup P_2)$, and the Minkowski sum $P_1 + P_2$. For Minkowski sum we prove that enumerating the facets of $P_1 + P_2$ is NP-hard if P_1 and P_2 are specified by facets, or if P_1 is specified by vertices and P_2 is a polyhedral cone specified by facets. For intersection we prove that computing the facets or the vertices of the intersection of two polytopes is NP-hard if one of them is given by vertices and the other by facets. Also, computing the vertices of the intersection of two polytopes given by vertices is shown to be NP-hard. Analogous results for computing the convex hull of the union of two polytopes follow from polar duality. All of the hardness results are established by showing that the appropriate decision version, for each of these problems, is NP-complete .

1 Introduction

A *convex polyhedron* or simply *polyhedron* in the d-dimensional Euclidean space $\mathbb{R}^d$ is the intersection of a finite number of halfspaces. A polyhedron is called *pointed* if it does not contain any affine line in its interior and *bounded* if it does not contain any ray. A bounded polyhedron is also called a polytope. A very basic result in the theory of polyhedra states that a polyhedron can be described both as the intersection of a finite number of halfspaces as well as the Minkowski sum of a polytope and a polyhedral cone. In other words every polyhedron can be represented as $conv(V) + cone(Y)$, where V and Y are finite sets of points in $\mathbb{R}^d$. The elements of V are called the vertices, and the elements of Y are called the extreme rays of the polyhedron. For any pointed polyhedron the minimal such representation is unique and we will always assume that V is minimal. If V contains just one point then we call the polyhedron a polyhedral cone or simply a cone. If $Y = \emptyset$, then the polyhedron is bounded and is a polytope. For a thorough treatment of the subject Grünbaum [10] and Ziegler [16] are excellent sources.

A polytope described by its vertices is called a $\mathcal{V}$-polytope and a polytope described by its facets is called an $\mathcal{H}$-polytope. Accordingly, we refer to the two equivalent representations as $\mathcal{V}$-representation and $\mathcal{H}$-representation respectively. Many operations that are easy to perform starting with one description become difficult if one starts with the other description. To give a simple example, finding a point inside a polytope that maximizes the inner product with a given vector can be done trivially if the polytope is in $\mathcal{V}$-representation but for the $\mathcal{H}$-representation this amounts to Linear Programming for which only weak polynomiality is known [13]. Weak polynomiality means that the number of arithmetic operations performed by the algorithm at hand is a polynomial in the number of bits needed to specify the input, as for example in Khachiyan's ellipsoid method for linear programming. In contrast, a strongly polynomial algorithm for linear programming would require a number of arithmetic opera-

tions that is a polynomial only in the number of constraints and the dimension of the linear program. For a formal definition of weak and strong polynomiality, the reader is referred to [9].

In this paper, we study three fundamental operations on polytopes and provide hardness results for them. For polytopes $P_1, P_2 \subset \mathbb{R}^d$, the Minkowski addition $P_1 + P_2$, the convex hull of the union $CH(P_1 \cup P_2)$ and the intersection $P_1 \cap P_2$ are defined as:

$$\begin{aligned} P_1 + P_2 &= \{x + y | x \in P_1, y \in P_2\} \\ CH(P_1 \cup P_2) &= \{\lambda x + (1-\lambda)y | x \in P_1, y \in P_2, 0 \leq \lambda \leq 1\} \\ P_1 \cap P_2 &= \{x | x \in P_1 \wedge x \in P_2\} \end{aligned}$$

We are interested in the complexity of performing these operations and providing non-redundant description of the resulting polytope in appropriate representation. Since the worst case size of the output for all the three operations can be exponential in the size of input (see [7, 8]), it is natural to talk of *output sensitive* algorithms. The complexity of an output-sensitive algorithm is measured in terms of the size of both the input and the output. Thus, a polynomial output-sensitive algorithm is one whose running time is polynomial in the size of the input and the output. In what follows, we will generally use the term "output-sensitive" to mean "polynomial output-sensitive".

It is easy to see that computing the non-redundant $\mathcal{V}$-representation of $CH(P_1 \cup P_2)$ and $P_1 + P_2$ is easy if P_1, P_2 are $\mathcal{V}$-polytopes, since redundancy in the output can be removed by solving a polynomial number of linear programs. Similarly, the non-redundant $\mathcal{H}$-representation of $P_1 \cap P_2$ can be computed via linear programming if P_1, P_2 are $\mathcal{H}$-polytopes. Therefore, we are interested in other versions of these problems. In this paper, for Minkowski sum we consider and prove hardness results for the version where P_1, P_2 are $\mathcal{H}$-polytopes or, where one is a $\mathcal{V}$-polytope while the other is a polyhedral cone given by its facets. In both cases we want to compute the $\mathcal{H}$-representation of the Minkowski

sum $P_1 + P_2$.

For $P_1 \cap P_2$ we consider and prove hardness results for the following three variants:

- Given two $\mathcal{V}$-polytopes P_1 and P_2, output the vertices of $P_1 \cap P_2$.
- Given an $\mathcal{H}$-polytope P_1 and a $\mathcal{V}$-polytope P_2, output the vertices of $P_1 \cap P_2$.
- Given an $\mathcal{H}$-polytope P_1 and a $\mathcal{V}$-polytope P_2, output the facets of $P_1 \cap P_2$.

Intersection and convex hull of the union are operations dual to each other and any statement about one can be translated into a similar statement about the other, via polar duality, with points replacing hyperplanes and vice-versa. Accordingly, for $CH(P_1 \cup P_2)$ we consider the versions analogous to that of the intersection, with $\mathcal{V}$-representation replaced by $\mathcal{H}$-representation and vice-versa. Before proceeding further we will describe the notion of polarity that relates these two operations. We also describe the Cayley embedding of polytopes which relates Minkowski addition with the convex hull of the union.

1.1 Polarity

Let $P = \{x | Ax \leq \mathbf{1}\}$ be a full dimensional polyhedron in $\mathbb{R}^d$ containing the origin in its relative interior. Here $A \in \mathbb{R}^{m \times d}$ is a matrix with m rows and d columns, and $\mathbf{1}$ is an $m \times 1$ column vector with all entries 1. The polar (also dual) of P, denoted by P^*, is obtained by treating the row vectors a_i of A as points in $\mathbb{R}^d$ and taking the convex hull of all these points together with the origin. If P is bounded then the origin lies in the relative interior of the polar. For a detailed treatment of this operation the reader is again referred to [10] and [16]. One interesting property of the polar operation is that a point α in the relative interior of P is mapped to the hyperplane $\{x | \alpha \cdot x = 1\}$ that does not intersect P^*. Similarly a point on the boundary of P is mapped to a hyperplane that touches P^* and a point outside P is mapped to a hyperplane that intersects the interior of P^*.

The convex hull of the union and the intersection operations are related via polar duality. More precisely, if P_1, P_2 are two full dimensional polytopes (or polyhedra) in $\mathbb{R}^d$ both containing origin in the interior, then $P_1 \cap P_2$ is the polar dual of $CH(P_1^* \cup P_2^*)$.

1.2 The Cayley trick

The Cayley trick ([11, 14]) allows us to represent the Minkowski sum $P_1 + P_2$ of two polytopes as the intersection of a $(d+1)$-polytope with a hyperplane. Consider two d-dimensional polytopes P_1 and P_2. Embed the two polytopes in $\mathbb{R}^{d+1}$ by putting a copy of P_1 in the hyperplane $\{x_{d+1} = -1\}$ and a copy of P_2 in the hyperplane $\{x_{d+1} = 1\}$. Let P be the $(d+1)$-dimensional polytope obtained by taking the convex hull of both embedded polytopes. Then the Minkowski sum (scaled by a factor half) of P_1 and P_2 is the intersection of P with the hyperplane $\{x_{d+1} = 0\}$.

1.3 Model of Computation and Complexity Notions

We will use a bit model of computation where the input is comprised of n rational numbers each requiring at most L bits in its binary representation. Thus, for the problems considered in this paper, we assume that the polytope is described by rational numbers encoding the dimension, the number of vertices (or facets) and a sequence of rational numbers encoding the coordinates of vertices, or the facet normals in case the polytope is given by $\mathcal{H}$-representation. The total number of bits required is called the *size* of the input polytope.

We will only count the number of arithmetic operations as the measure of the running time of an algorithm and accordingly, we will call an algorithm polynomial if the number of arithmetic operations performed by the algorithm is a polynomial in n and L. For all our reductions, the number of bits L in the binary representation of input polytopes is a polynomial in the number of vertices and the ambient dimension of the polytope. Hence, our hardness results

imply that all the problems shown to be NP-hard in this paper are *strongly* NP-hard.

The rest of the paper is organized as follows. In the next section, we describe prior work related to performing these operations in appropriate representations and in Section 3 we describe the hardness results for computing the Minkowski sum of two polytopes. In Section 4 we establish hardness results for computing the intersection of two polytopes in various representations.

2 Related Work

The problem of enumerating the facets of $CH(P_1 \cup P_2)$, when both P_1 and P_2 are given by their facets, has been studied in [2] and [5]. Balas [2] constructs polynomial algorithm for a special class of polytopes arising in 0-1 Mixed Integer programming, while Fukuda, Liebling and Lütolf [5] present an algorithm that has polynomial complexity if the input polytopes satisfy certain general position assumptions. It is not clear if arbitrary polytopes can be made to satisfy the general position assumption as described in [5]. The NP-hardness of computing the convex hull of the union of polytopes, as proved in this paper, suggests that these assumptions are probably unrealistic for general polytopes.

Minkowski sums have been studied much more compared to the convex hull of the union. They frequently come up in computational algebra [8], robotics and motion planning, geometric convexity, computer graphics and many other areas. Gritzmann and Sturmfels [8] studied Minkowski sum in the context of computational algebra and gave (exponential) bounds on the number of faces of the Minkowski sum. They also gave examples of cases where the bounds are tight.

As noted before, exponential lower bounds motivate one to look for output sensitive algorithms so that cases where the output is far from worst case can be handled efficiently. Komei Fukuda [4] proposed a polynomial algorithm for enumerating all vertices of the Minkowski sum of k $\mathcal{V}$-polytopes. In a subsequent

paper Fukuda and Weibel [6] gave a polynomial algorithm for enumerating all faces of k $\mathcal{V}$-polytopes. They do not consider the case when the input polytopes are described by facets and the facets of Minkowski sum are to be enumerated, and note this version of the problem to be open. Fukuda and Weibel, in another work (see [7]), study Minkowski sums of special polytopes that are "well centered" and also provide better bounds on the number of faces for this special case.

Our main results state that the following decision problems are NP-complete and thus there is no output-sensitive algorithm for the corresponding enumeration problems unless $P = NP$:

- Given an $\mathcal{H}$-polytope P_1, an $\mathcal{H}$-polytope P_2, and an $\mathcal{H}$-polytope Q, is $P_1 + P_2 \neq Q$?
- Given an $\mathcal{H}$-cone P_1, a $\mathcal{V}$-polytope P_2, and an $\mathcal{H}$-polyhedron Q, is $P_1 + P_2 \neq Q$?
- Given a $\mathcal{V}$-polytope P_1, a $\mathcal{V}$-polytope P_2, and a $\mathcal{V}$-polytope Q, is $P_1 \cap P_2 \neq Q$?
- Given an $\mathcal{H}$-polytope P_1, a $\mathcal{V}$-polytope P_2, and a $\mathcal{V}$-polytope Q, is $P_1 \cap P_2 \neq Q$?
- Given an $\mathcal{H}$-polytope P_1, a $\mathcal{V}$-polytope P_2, and an $\mathcal{H}$-polytope Q, is $P_1 \cap P_2 \neq Q$?

For the first decision problem we provide a Turing reduction from another NP-complete problem. Usually reductions for proving NP-completeness employ Karp reduction. A problem $\mathcal{A}$ is said to be polynomial-time Turing reducible to problem $\mathcal{B}$ if one can construct a polynomial time algorithm for problem $\mathcal{A}$ using an oracle for $\mathcal{B}$. The more common Karp reduction allows only one call to the oracle and that too at the end. For all other decision problems we provide the standard Karp reduction from some other NP-complete problem.

3 Hardness of Minkowski Addition

In this section we establish two hardness results about computing the facets of the Minkowski sum of two polytopes. We begin by proving the hardness of enumerating the facets of the Minkowski sum of two $\mathcal{H}$-polytopes. Consider the following decision version of the enumeration problem:

PROBLEM INCOMPLETEMINKOWSKI

INPUT: $\mathcal{H}$-Polytopes P_1, P_2, Q.

OUTPUT: Yes, if $Q \neq P_1 + P_2$. No, otherwise.

Theorem 1 INCOMPLETEMINKOWSKI *is NP-complete.*

It was shown by Khachiyan *et. al* [12] that it is NP-Hard to enumerate all vertices of a polyhedron given by its facets. The following theorem restates the result of [12].

Theorem 2 *Given a polyhedron P in $\mathcal{H}$-representation and a set V of vertices of P, it is NP-complete to determine if P has some vertex* not *in V.*

Now, we prove that if we have an algorithm for deciding INCOMPLETE-MINKOWSKI for arbitrary input polytopes, then we can invoke this oracle a polynomial number of times and decide for some set of vertices V and an $\mathcal{H}$-polyhedron P, whether $V \subset vert(P)$.

Let $P = \{x | Ax \leq b\}$ be a polyhedron in $\mathbb{R}^d$ and $V \subseteq vert(P)$ with $|V| = n$. We want to determine whether $V \subset vert(P)$ using a polynomial number of calls to an oracle for INCOMPLETEMINKOWSKI. For this we pick some direction and order the vertices of V in that direction. We assume that this direction is aligned with the x_d coordinate axis (after possibly applying a suitable affine transform). That is, if $\mathbf{e}_d$ is the unit vector $(0, \cdots, 0, 1)$ in $\mathbb{R}^d$ and $\mathbf{e}_d$ is thought to be the *upward* direction, then the vertices are considered in the order of increasing height, *i.e.* v_1 is the lowest vertex and v_n is the highest vertex. We also assume that any horizontal slice of P *i.e.* $P \cap \{x | x_d = c\}$ is a bounded polytope for any $c \in \mathbb{R}$. We justify this assumption later.

Now, consider vertices v_i and v_{i+1} for some fixed vertex subscript i and define three polytopes in the following way:

$$\begin{aligned} P_{-1} &= P \cap \{x_d = v_i \cdot \mathbf{e}_d\} \\ P_1 &= P \cap \{x_d = v_{i+1} \cdot \mathbf{e}_d\} \\ P_0 &= P \cap \left\{ x_d = \frac{v_i \cdot \mathbf{e}_d + v_{i+1} \cdot \mathbf{e}_d}{2} \right\} \end{aligned}$$

where the dot product $v_i \cdot \mathbf{e}_d$ is nothing but the x_d-coordinate (height) of v_i. Informally speaking, we are interested in three slices of the polyhedron: a top slice at the height of v_{i+1}, a bottom slice at the height of v_i and a slice at an intermediate height.

We claim that the middle slice is the Minkowski sum of the top and the bottom slices (with a scaling of half) if and only if there is no other (missing) vertex of P lying at an intermediate height between v_i and v_{i+1}. The following lemma states this formally.

Lemma 1 *$2P_0 \neq P_{-1} + P_1$ if and only if there exists some $v \in vert(P)$ that is not in V and $v_i \cdot \mathbf{e}_d < v \cdot \mathbf{e}_d < v_{i+1}.\mathbf{e}_d$.*

Proof: We prove the non-trivial direction only. Suppose some vertex $v \in vert(P)$ is not in V and $v_i \cdot \mathbf{e}_d < v \cdot \mathbf{e}_d < v_{i+1}.\mathbf{e}_d$ for the vertex subscript i under consideration. Without loss of generality we can assume that v lies above the hyperplane containing P_0. If so, there is an $u \in vert(P_{-1})$ such that $\overrightarrow{uv}$ lies on some edge of P. Clearly, $\overrightarrow{uv}$ intersects P_0, say at w. We claim that $2w \notin P_{-1} + P_1$.

Assume for the sake of contradiction that $2w \in P_{-1} + P_1$. Then there are $x \in P_{-1}$ and $y \in P_1$ such that $2w = x + y$. Since, any point on an edge of a polytope can be *uniquely* represented as the convex combination of the vertices defining the edge, it follows that $x = u$ and y is a vertex of P_1. This implies that v is a convex combination of x, y as well and hence, v can not be a vertex of P, a contradiction. □

To complete our algorithm for determining whether a given set V of vertices of an $\mathcal{H}$-polyhedron P is the complete vertex set of P, we also need to check the region below the lowest known vertex and above the highest known vertex. Thus, to complete the proof of Theorem 1 we need to be able to pick the direction "up" satisfying the following requirements:

(i) Every slice of P orthogonal to this direction is a polytope, *i.e* it is bounded.

(ii) Vertices of P have a unique ordering according to their heights in the "upward" direction. In other words, no two vertices of P have the same height.

(iii) We can find an upper bound on the height of all vertices of the polyhedron P. Furthermore, we require that this height be represented using a number of bits that is polynomial in the size of the input.

Note that for a direction satisfying requirements (i) and (ii), we can assume that the lowest vertex in the known set of V is also the lowest vertex of the polyhedron P. Also such a vertex can be found by solving a linear program and if V does not contain this vertex then clearly $V \subset vert(P)$. Furthermore, requirement (iii) allows us to check the region above the highest known vertex of P. We would also like that the direction satisfying the above requirements be represented using number of bits q that is polynomial in the number of bits used in the description of P and V.

3.1 Finding the sweep direction

To satisfy the first requirement, recall that a pointed polyhedron has a unique minimal representation as the Minkowski sum of a polytope and a cone. Also, the cone of the polyhedron $P = \{x|Ax \leq 1\}$ is just $cone(P) = \{x|Ax \leq 0\}$ with some inequalities possibly redundant. So a vector α such that $cone(P) \cap \{\alpha \cdot x \leq 1\}$ is bounded, satisfies the first requirement. Any vector picked from the interior $cone(A)$ does the trick, where every row of A is interpreted as a vector in $\mathbb{R}^d$. In particular the average of the row vectors of A satisfies requirement (i) and requires a number of bits that is polynomial in the size of A.

Let N be the set of the facet normals of $cone(A)$. Computing N is not an easy task but for our purposes we only need an upper bound on the size of the coefficients of these facet normals. It is known that the number of bits required to represent N is a polynomial in the number of bits required to represent A (See [9] Page 164, Lemma 6.2.4). An immediate consequence of the same Lemma

is that for any polyhedron $P = \{x|Ax \leq 1\}$, the number of bits required to represent the vertices of P is a polynomial in A, and hence a polynomial upper bound on the height of the topmost vertex can be computed. Thus assumption (iii) can be satisfied as long as the sweep direction needs a polynomial number of bits in its representation.

The first two conditions, for any possible sweep direction a, can be rewritten as:

$$\langle a, \eta \rangle < 0, \qquad \forall \eta \in N \tag{1}$$

$$\langle a, u - v \rangle \neq 0, \quad \forall u, v \in vert(P), u \neq v \tag{2}$$

where $\langle x, y \rangle$ is the inner product of the vectors x and y.

We already have a direction α satisfying (1). Now, consider the directions

$$\begin{aligned} \beta &= \begin{pmatrix} 1 \\ x \\ \vdots \\ x^{d-1} \end{pmatrix} \\ \gamma &= x^d \alpha + \beta \end{aligned}$$

We show that for large enough x, the direction γ satisfies both the requirements (1) and (2), and that the number of bits needed for x, and hence for γ, is a polynomial in the size of the polyhedron P. We want that for γ

$$\langle \gamma, \eta \rangle = x^d \langle \alpha, \eta \rangle + \langle \beta, \eta \rangle < 0, \quad \forall \eta \in N \tag{3}$$

$$\langle \gamma, u - v \rangle = x^d \langle \alpha, u - v \rangle + \langle \beta, u - v \rangle \neq 0, \quad \forall u, v \in vert(P), u \neq v \tag{4}$$

Notice that equations (3) and (4) involve polynomials in x whose coefficients depend only on α, N and $vert(P)$. Also, recall that the sizes of α, N and $vert(P)$ are each a polynomial in the size of the input polytope, i.e. the size of A, and

so the size of the coefficients in the polynomials involved in equation (3) and (4) is a polynomial in the size of the input polyhedron. For large enough x the sign of the polynomial in equation (3) is the same as the sign of $\langle \alpha, \eta \rangle$. Since α satisfies (1), γ satisfies (1) as well for large enough x. It is also clear that the size of such an x need only be a polynomial in the size of the coefficients of the polynomial in (3).

Also, any polynomial in x evaluates to a non-zero value if x is larger than the largest possible root of the polynomial. Since the largest root of a polynomial has size polynomial in the size of its coefficients (See [15], page 148, Lemma 6.7), the size of x required to satisfy equation 4 and hence condition (2) is a polynomial in the size of the coefficients involved in 4. This proves that we can pick a direction γ satisfying all the necessary conditions and requiring a number of bits that is polynomial in the size of P.

Thus, the polyhedron P and the vertex list V can be preprocessed so that their sizes remain polynomial, and so that if $V \subset vert(P)$ then Lemma 1 can be used to find a missing vertex by checking the space between v_i and v_{i+1} for each i, and checking the space above the highest vertex. As stated before, any vertex of P requires a number of bits that is bounded by a polynomial in the size of P and so we can check the region above the highest vertex as well.

The above reduction proves that IncompleteMinkowski is NP-hard. To prove that IncompleteMinkowski is in NP as well, notice that given a hyperplane $h : \{a \cdot x = 1\}$ one can easily check whether it defines a facet of the Minkowski sum $P_1 + P_2$. To see this, let us consider the Cayley embedding of the two polytopes. If this hyperplane defines a facet of the Minkowski sum $P_1 + P_2$, then in the Cayley embedding as well, it corresponds to a facet of the convex hull of the union of the two polytopes. Given a hyperplane h one can find the faces of P_1 and P_2 that (possibly) define the corresponding facet of the Cayley embedding. For each P_i this can be done by simply translating the hyperplane away from the origin until it becomes a supporting hyperplane for P_i and taking the face of P_i contained in the hyperplane at this point. Whether these two faces

define a facet of the Cayley embedding or not can be checked by just checking the dimension of the convex hull of the union of these two faces. This completes the proof of Theorem 1.

Following is an immediate corollary of Theorem 1:

Corollary 1 *Given two $\mathcal{H}$-polytopes $P_1, P_2 \in \mathbb{R}^d$, there is no output-sensitive algorithm that enumerates the facets of $P_1 + P_2$ unless $P = NP$.*

Since for an $\mathcal{H}$-polyhedron P and a subset of its vertices V, it is NP-complete to decide whether $V \subset vert(P)$, we also have the following theorem:

Theorem 3 *Given an $\mathcal{H}$-cone P_1, a $\mathcal{V}$-polytope P_2, and an $\mathcal{H}$-polyhedron Q, it is NP-complete to determine whether $P_1 + P_2 \neq Q$.*

Proof: Any pointed polyhedron P has a unique minimal representation as the Minkowski sum of the polytope defined by its vertices and the cone of its extreme rays. Also, the cone of the extreme rays of the polyhedron $P = \{x | Ax \leq b\}$ is just $cone(P) = \{x | Ax \leq 0\}$. Redundant inequalities of $cone(P)$ can be removed using linear programming and hence $conv(V) + cone(P) = P$ if and only if $V = vert(P)$. Thus, an algorithm for enumerating the facets of the Minkowski sum of a $\mathcal{V}$-polytope and an $\mathcal{H}$-cone can be used to determine whether a given list of vertices of a polyhedron P is complete or not.

As noted earlier, given a hyperplane it can be easily checked whether it defines a facet of the Minkowski sum of P_1 and P_2. Thus, if $P_1 + P_2 \neq Q$ then there is a facet defining hyperplane for $P_1 + P_2$ that does not define a facet of Q. This proves that this decision problem is in NP as well. □

It should be remarked that if the cone P_1 in Theorem 3 is represented by it's extreme rays then the problem of determining whether $P_1 + P_2 = Q$ or not, is equivalent to the problem of computing the V-representation of a polytope from the H-representation and vice-versa. The complexity status of the representation conversion problem remains open despite years of research [1].

4 Hardness of Computing Intersection

Recall that the Minkowski sum of two polytopes can be computed via computing the convex hull of two polytopes using the Cayley embedding. To compute the Minkowski sum of polytopes P and Q in $\mathbb{R}^d$, we embed the polytopes in $\mathbb{R}^{d+1}$ by putting a copy of P in the hyperplane defined by $\{x_{d+1} = -1\}$ and a copy of Q in the parallel hyperplane $\{x_{d+1} = 1\}$. If P_{-1} and Q_1 are the copies of P and Q respectively, then $P + Q$ is obtained (upto a scaling factor $\frac{1}{2}$) by taking the convex hull $CH(P_{-1} \cup Q_1)$ and intersecting it with the hyperplane $\{x_{d+1} = 0\}$.

Note that, the operand polytopes P_{-1} and Q_1 here are not full dimensional *i.e.* even though they are embedded in $\mathbb{R}^{d+1}$, neither of them has dimension $d + 1$. However, one can easily ensure that these polytopes are full dimensional and both contain the origin in their relative interiors. To do this, we pick a point p in the relative interior of $CH(P_{-1} \cup Q_1)$ and construct a pyramid with base P_{-1} and p as the apex. It is easy to see that this can be done in polynomial time. Now we can pick another point q in the relative interior of this pyramid and create a pyramid with Q_1 as the base and q as the apex. Since the convex hull of the union of these two pyramids is the same as that of P_{-1} and Q_1 and their intersection is a full dimensional polytope, we can move origin in this common region. This together with Theorem 3 gives us the following theorem:

Theorem 4 *Given $\mathcal{H}$-polytopes $P_1, P_2, Q \in \mathbb{R}^d$, it is NP-complete to decide whether $CH(P_1 \cup P_2) \neq Q$.*

This can be dualized since each of the polytopes is full dimensional and contains origin in the relative interior. By considering the polar duals of P_1, P_2 and Q, each of which is a full dimensional $\mathcal{V}$-polytope containing the origin in the relative interior, we have the following theorem:

Theorem 5 *Given $\mathcal{V}$-polytopes $P_1, P_2, Q \in \mathbb{R}^d$, it is NP-complete to decide whether $P_1 \cap P_2 \neq Q$.*

Now we prove that the problem of computing either the facets or the ver-

tices of the intersection of two polytopes is hard for the case where one of the polytopes is given by $\mathcal{H}$-representation and the other by $\mathcal{V}$-representation.

Theorem 6 *Given an $\mathcal{H}$-polytope P_1, a $\mathcal{V}$-polytope P_2, and an $\mathcal{H}$-polytope Q, it is NP-complete to decide whether $P_1 \cap P_2 \neq Q$.*

Proof: It is known ([3]) that given an $\mathcal{H}$-polytope P_1 and a $\mathcal{V}$-polytope P_2, it is NP-complete to decide whether $P_1 \nsubseteq P_2$. Clearly, $P_1 \subseteq P_2$ if and only if $P_1 \cap P_2 = P_1$. This implies that checking whether a given list of facets completely defines the intersection of an $\mathcal{H}$-polytope and a $\mathcal{V}$-polytope, is NP-hard.

The problem is also in NP because for given polytopes P_1, P_2 and Q, if $P_1 \cap P_2 \neq Q$ then Q has a vertex that does not lie in the intersection $P_1 \cap P_2$. A point lies in $P_1 \cap P_2$ if and only if it satisfies all the facet inequalities of P_1 and can be represented as the convex combination of the vertices of P_2. Both the tests can be performed in polynomial time for rational polytopes. □

As it turns out computing the vertices of the intersection of an $\mathcal{H}$-polytope and a $\mathcal{V}$-polytope is hard as well.

We know from Theorem 3 that it is NP-complete to decide whether a given list of facets of the Minkowski sum of an $\mathcal{H}$-cone P_1 and a $\mathcal{V}$-polytope P_2 in $\mathbb{R}^d$, is complete or not. As stated in the beginning of this section, we can embed P_1 and P_2 in $\mathbb{R}^{d+1}$ in two parallel hyperplanes and the Minkowski sum $P_1 + P_2$ is the intersection of the convex hull of P_1 and P_2 with an appropriate hyperplane. Also, we can pick points p and q in the convex hull of P_1 and P_2 such that the pyramids P_1' and P_2' obtained from P_1 with apex p and P_2 with apex q are full dimensional and have a full dimensional intersection. Thus,

Theorem 7 *Given an $\mathcal{H}$-polyhedron P_1 and a $\mathcal{V}$-polytope P_2, it is NP-hard to compute the facets of the polyhedron $CH(P_1 \cup P_2)$.*

Consider the polar duals of P_1 and P_2. Since both P_1 and P_2 are full dimensional and contain the origin in their relative interiors, the polar dual of P_2 is bounded *i.e.* an $\mathcal{H}$-polytope, and the polar dual of P_1 is a $\mathcal{V}$-polytopes with

vertices $A \cup \{0\}$ if P_1 is represented as $Ax \leq \mathbf{1}$. The vertices of $P_1^* \cap P_2^*$ are in one-to-one correspondence with the facets of $CH(P_1 \cup P_2)$ and so we have the following theorem:

Theorem 8 *Given an $\mathcal{H}$-polytope P_1, a $\mathcal{V}$-polytope P_2 and a $\mathcal{V}$-polytope Q, it is NP-complete to decide whether $P_1 \cap P_2 \neq Q$.*

5 Acknowledgements

The author was supported by Graduiertenkolleg fellowship for PhD studies provided by Deutsche Forschungsgemeinschaft when some of this research was done. The author would also like to thank Günter Rote for helpful discussions relating to Subsection 3.1, and Raimund Seidel for extremely helpful comments regarding the content and the structure of this paper, and also for suggesting simplifications to the discussions in Subsection 3.1.

References

[1] D. Avis, D. Bremner, and R. Seidel. How good are convex hull algorithms? *Comput. Geom.*, 7:265–301, 1997.

[2] E. Balas. On the convex hull of the union of certain polyhedra. *Operations Research Letters*, 7:279–283, 1988.

[3] R. M. Freund and J. B. Orlin. On the complexity of four polyhedral set containment problems. *Mathematical Programming*, 33(2):139–145, 1985.

[4] K. Fukuda. From the zonotope construction to the minkowski addition of convex polytopes. *J. Symb. Comput.*, 38(4):1261–1272, 2004.

[5] K. Fukuda, T. M. Liebling, and C. Lutolf. Extended convex hull. *Computational Geometry*, 20(1-2):13–23, 2001.

[6] K. Fukuda and C. Weibel. Computing all faces of the minkowski sum of $\mathcal{V}$-polytopes. *In Proceedings of the 17th Canadian Conference on Computational Geometry*, 2005.

[7] K. Fukuda and C. Weibel. f-vectors of Minkowski additions of convex polytopes. *Discrete Comput. Geom.*, 37:503–516, 2007.

[8] Gritzmann and Sturmfels. Minkowski addition of polytopes: Computational complexity and applications to grobner bases. *SIJDM: SIAM Journal on Discrete Mathematics*, 6, 1993.

[9] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, 1993.

[10] B. Grünbaum. *Convex Polytopes, Second Edition prepared by V. Kaibel, V. L. Klee and G. M. Ziegler*, volume 221 of *Graduate Texts in Mathematics*. Springer, 2003.

[11] B. Huber, J. Rambau, and F. Santos. The cayley trick, lifting subdivisions and the bohne-dress theorem on zonotopal tilings. *Journal of the European Mathematical Society*, 2(2):179–198, 2000.

[12] L. Khachiyan, E. Boros, K. Borys, K. M. Elbassioni, and V. Gurvich. Generating all vertices of a polyhedron is hard. In *SODA*, pages 758–765. ACM Press, 2006.

[13] L. G. Khachiyan. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[14] B. Strumfels. On the newton polytope of the resultant. *Journal of Algebraic Combinatorics*, 3(2):207–236, 1994.

[15] C. K. Yap. *Fundamental Problems in Algorithmic Algebra*. Oxford University Press, New York, 2000.

[16] G. M. Ziegler. *Lectures on Polytopes*. Graduate Texts in Mathematics, No. 152. Springer-Verlag, Berlin, 1995.