An Implementation of the Barvinok–Woods Integer Projection Algorithm

Matthias Köppe
Dept. of Mathematics / IMO
Universität Magdeburg
Universitätsplatz 2
39106 Magdeburg, Germany
Email: mkoeppe@ovgu.de

Sven Verdoolaege[*]
LIACS
Universiteit Leiden
Niels Bohrweg 1
2333 CA Leiden, The Netherlands
Email: sverdool@liacs.nl

Kevin M. Woods
Dept. of Mathematics
Oberlin College
Oberlin, OH, USA
Email: Kevin.Woods@oberlin.edu

Abstract ***We describe the first implementation of the Barvinok–Woods (2003) algorithm, which computes a short rational generating function for an integer projection of the set of integer points in a polytope in polynomial time, when the dimension is fixed. The algorithm is based on Kannan's partitioning lemma and the application of set operations to generating functions that correspond to these sets. We use a variant of the recent strengthening of the partitioning lemma due to Eisenbrand and Shmonin (2007) and provide several algorithmic refinements to avoid performing redundant set operations. The implementation has been done in the second author's library*** **barvinok.**

Keywords: integer hulls, integer projections, parametric polytopes, partitioning lemma, rational generating functions

1 Introduction

Barvinok [1] introduced an algorithm for determining the exact number of lattice points in a rational polytope $P \subseteq \mathbb{R}^n$ that runs in polynomial time for every fixed dimension n. The algorithm computes a representation of the *generating function*

$$g(P; \boldsymbol{\xi}) = \sum_{\mathbf{x} \in P \cap \mathbb{Z}^n} \boldsymbol{\xi}^{\mathbf{x}} = \sum_{\mathbf{x} \in P \cap \mathbb{Z}^n} \xi_1^{x_1} \cdots \xi_n^{x_n} \qquad (1.1)$$

as a rational function of polynomial encoding size (in fixed dimension). Such *short rational generating functions* of lattice point sets can be manipulated efficiently. For example, residue techniques can be used to evaluate $g(P; \mathbf{1})$, which provides the number of lattice points of P. Also, Barvinok and Woods [2] proved that arbitrary constant-length Boolean combinations of finite lattice point sets given by rational generating functions can be constructed in polynomial time (when the dimension is fixed).

The topic of this paper is a more general construction also due to Barvinok and Woods [2]. Let $P \subseteq \mathbb{R}^n$ be a polytope. Let $\pi\colon \mathbb{Z}^n \to \mathbb{Z}^k$ be a $\mathbb{Z}$-linear map and denote by $S = \pi(P \cap \mathbb{Z}^n)$ the image (projection) of $P \cap \mathbb{Z}^n$. Barvinok and Woods [2, Theorem 1.7] proved that a short rational generating function for this projection can be constructed in polynomial time, when the dimension is fixed:

Theorem 1.2 (Projection Theorem) *Let the dimension n be fixed. Then there exist a constant $s(n)$ and a polynomial-time algorithm for the following problem. Given as* input*, in binary encoding,*

(I$_1$) *inequalities describing a rational polytope $P \subset \mathbb{R}^n$,*

(I$_2$) *a linear map $\pi\colon \mathbb{Z}^n \to \mathbb{Z}^k$,*

output*, in binary encoding,*

(O$_1$) *rational numbers γ_i, integer vectors $\mathbf{c}_i$, $\mathbf{d}_{ij}$ for $i \in I$, $j = 1, \ldots, s_i$, where $s_i \le s(n)$,*

[*]Part of this work was completed at the Dept. of Computer Science, Katholieke Universiteit Leuven

such that a rational generating function of the set $S = \pi(P \cap \mathbb{Z}^n)$ *is given by*

$$g(S; \boldsymbol{\xi}) = \sum_{i \in I} \gamma_i \frac{\boldsymbol{\xi}^{\mathbf{c}_i}}{(1-\boldsymbol{\xi}^{\mathbf{d}_{i1}}) \dots (1-\boldsymbol{\xi}^{\mathbf{d}_{is_i}})}. \quad (1.3)$$

This projection theorem has many important consequences in integer programming [3], multi-objective integer linear programming [4], bilevel integer programming and algorithmic game theory [5]. To the best of our knowledge, no implementation of the Barvinok–Woods projection algorithm has appeared in the literature. Because the algorithm contains complicated constructions such as algorithmic flatness theory and iterated Boolean combinations of rational generating functions, the algorithm has frequently been referred to as "unimplementable".

This paper attempts to change that. We report on the first-ever implementation of the algorithm, within the second author's library *barvinok* [6]. The implementation makes use of several refinements of the algorithm, which we also discuss in the paper.

2 The Barvinok–Woods Integer Projection Algorithm

We discuss the algorithm in a more general, parametric setting introduced by [7] that deals with counting functions

$$c(\mathbf{s}) = \# \left\{ \mathbf{t} \in \mathbb{Z}^d \mid \exists \mathbf{u} \in \mathbb{Z}^m : (\mathbf{s}, \mathbf{t}, \mathbf{u}) \in P \right\},$$

where $P \subset \mathbb{Q}^{n+d+m}$ is a rational pointed polyhedron such that $P_\mathbf{s} = \left\{ (\mathbf{t}, \mathbf{u}) \in \mathbb{Q}^{d+m} \mid (\mathbf{s}, \mathbf{t}, \mathbf{u}) \in P \right\}$ is bounded for any $\mathbf{s}$. (Note that, in contrast to [2], we may allow P itself to be unbounded here.) For details we refer to [7] and simply mention that we need to compute a rational generating function $f(\mathbf{x}, \mathbf{y})$ for the projection

$$T = \{ (\mathbf{s}, \mathbf{t}) \in \mathbb{Z}^{n+d} \mid \exists \mathbf{u} \in \mathbb{Z}^m : (\mathbf{s}, \mathbf{t}, \mathbf{u}) \in P \},$$

and then $\sum_\mathbf{s} c(\mathbf{s}) \, \mathbf{x}^\mathbf{s} = f(\mathbf{x}, \mathbf{1})$. If there is only one existentially quantified variable ($m = 1$), then computing T is easy: We shift P by 1 in the u direction and subtract this shifted copy from the original,

$$D = (P \cap \mathbb{Z}^{n+d+m}) \setminus \left(\mathbf{e}_{n+d+1} + (P \cap \mathbb{Z}^{n+d+m}) \right).$$

In the difference D there will be *exactly* one value of u for each value of the remaining variables for which there was *at least* one value of u in P. Thus the projection from D onto T is one-to-one. All of this can then be implemented on the level of rational generating functions, using Boolean operations and monomial substitution.

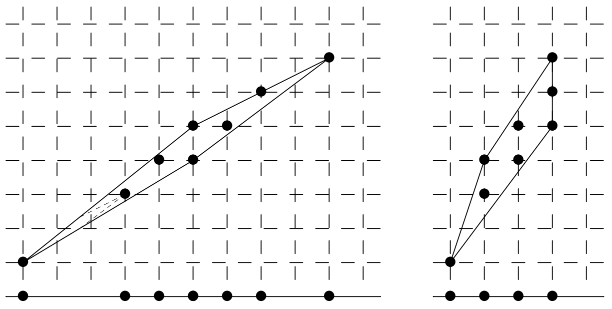
Figure 2.1: A polytope and its integer projections; the same after unimodular transformation

If there is more than one existentially quantified variable ($m > 1$), then we can in principle apply this technique recursively. A complication is that, after applying the technique in one direction, the resulting set S can contain "gaps". In Figure 2.1, imagine $m = 2$ and we want to project down to a single point. After projecting onto the horizontal direction, the biggest gap is 3, so we need to compute

$$S \setminus (\mathbf{e}_{n+d+1} + S) \setminus (2\mathbf{e}_{n+d+1} + S) \setminus (3\mathbf{e}_{n+d+1} + S).$$

In general, there is no bound on the widths of the gaps we may encounter. However, as a consequence of *flatness theory* for lattice-point-free convex bodies, it is possible to *construct* directions of small gaps for the parametric polytopes $P_{\mathbf{s},\mathbf{t}}$. We first need to introduce the notion of *lattice width.*

Definition 2.2 *We define the* width *of the polytope* $P_{\mathbf{s},\mathbf{t}}$ *in an integer direction* $\mathbf{c} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ *as*

$$\operatorname{width}_\mathbf{c} P_{\mathbf{s},\mathbf{t}} = \max_{\mathbf{x} \in P_{\mathbf{s},\mathbf{t}}} \langle \mathbf{c}, \mathbf{x} \rangle - \min_{\mathbf{x} \in P_{\mathbf{s},\mathbf{t}}} \langle \mathbf{c}, \mathbf{x} \rangle. \quad (2.3)$$

The lattice width *is the minimum width:*

$$\operatorname{width} P_{\mathbf{s},\mathbf{t}} = \min_{\mathbf{c} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}} \operatorname{width}_\mathbf{c} P_{\mathbf{s},\mathbf{t}}. \quad (2.4)$$

The following "small-gaps theorem" is a version of Theorem 4.3 in [2].

Theorem 2.5 *Let* $\kappa \geq 1$ *and let* $\mathbf{c} \in \mathbb{Z}^m$ *be a* κ*-*approximative lattice width direction*, i.e.,*

$$\operatorname{width}_\mathbf{c} P_{\mathbf{s},\mathbf{t}} \leq \kappa \cdot \operatorname{width} P_{\mathbf{s},\mathbf{t}}.$$

Then the image $Y = \{ \langle \mathbf{c}, \mathbf{x} \rangle \mid \mathbf{x} \in P_{\mathbf{s},\mathbf{t}} \cap \mathbb{Z}^d \}$ *does not have gaps larger than* $\kappa \cdot \omega(m)$.

Here $\omega(m)$ is the *flatness constant*; it only depends on the dimension m [8, 9, 10]. Kannan [11] proved the following theorem:

Theorem 2.6 (Kannan) *Let the total dimension $n + d + m$ be fixed. Then there exists a polynomial-time algorithm for the following problem. Given as* input*, in binary encoding,*

(I$_1$) *integers n, d, m,*

(I$_2$) *inequalities describing a rational polytope $P \subset \mathbb{R}^{n+d+m}$,*

output*, in binary encoding,*

(O$_1$) *inequality systems describing partially open polyhedra $\tilde{Q}_1, \dots, \tilde{Q}_M \subset \mathbb{R}^{n+d}$ that form a partition of the projection, Q, of P onto the first $n + d$ coordinates,*

(O$_2$) *integer vectors $\mathbf{c}_1, \dots, \mathbf{c}_M \in \mathbb{R}^m$,*

such that $\mathbf{c}_i$ is a 2-approximative lattice width direction for every polytope $P_{\mathbf{s},\mathbf{t}}$ when $(\mathbf{s}, \mathbf{t}) \in \tilde{Q}_i$.

(We will discuss a stronger version of Theorem 2.6, due to [12], in section 3.)

Now let $P_i = P \cap (\tilde{Q}_i \times \mathbb{Q}^m)$ and let $f_i(\mathbf{x}, \mathbf{y})$ be the generating function of the set

$$T_i = \{(\mathbf{s}, \mathbf{t}) \mid \exists \mathbf{u} \in \mathbb{Z}^m : (\mathbf{s}, \mathbf{t}, \mathbf{u}) \in P_i\}.$$

Then clearly, $f(\mathbf{x}, \mathbf{y}) = \sum_i f_i(\mathbf{x}, \mathbf{y})$. From now on, we will consider a particular P_i with corresponding κ-approximative lattice width direction $\mathbf{c}_i$ and drop the i subscript. We are thus given a polyhedron P such that a κ-approximative lattice width direction of $P_{\mathbf{s},\mathbf{t}}$ is $\mathbf{c}$. Without loss of generality, $\mathbf{c}$ is primitive, so we can extend the row vector $\mathbf{c}^T$ to a unimodular matrix, which we use to transform the $\mathbf{u}$ variables in the polyhedron P; see Figure 2.1. Using the notations $\mathbf{u}'$ and P', we have

$$T = \{(\mathbf{s}, \mathbf{t}) \mid \exists \mathbf{u}' \in \mathbb{Z}^m : (\mathbf{s}, \mathbf{t}, \mathbf{u}') \in P'\},$$

i.e., we have changed the values of the existentially quantified variables, but we have not changed the set T. Now consider the set

$$T' = \{(\mathbf{s}, \mathbf{t}, u'_1) \mid \exists u'_2, \dots, u'_m \in \mathbb{Z} : (\mathbf{s}, \mathbf{t}, \mathbf{u}') \in P'\}.$$

This set has only $m - 1$ existentially quantified variables, so we apply the projection algorithm recursively and obtain the generating function $f'(\mathbf{x}, \mathbf{y}, z)$ for T'. By construction, the gaps in the final coordinate of T' are small ($\leq \kappa \cdot \omega(m)$). We now compute the generating function $f(\mathbf{x}, \mathbf{y})$ for T. By computing

$$f''(\mathbf{x}, \mathbf{y}, z) = f'(\mathbf{x}, \mathbf{y}, z) \bigodot_{k=1}^{\lfloor \kappa\omega(m) \rfloor} \left(z^k f'(\mathbf{x}, \mathbf{y}, z)\right), \tag{2.7}$$

where $\odot$ is the set-difference operation on generating functions, we obtain a generating function for a set T'' where only the smallest value of u'_1 is retained. Thus the projection from T'' onto T is actually one-to-one, and we have $f(\mathbf{x}, \mathbf{y}) = f''(\mathbf{x}, \mathbf{y}, 1)$.

3 The Eisenbrand–Shmonin Variant of Kannan's Partitioning

In computational experiments we observed that the number of set-difference operations in (2.7), $\lfloor \kappa \cdot \omega(m) \rfloor$, has a great influence on the complexity of the resulting formulas and the running time. Since the number $\omega(m)$ is a fixed constant from flatness theory, the question arises whether there is a way to improve the approximation guarantee κ. In fact, there is — in a recent work, Eisenbrand and Shmonin [12] showed that it is actually possible, in polynomial time in fixed dimension, to compute a partition of the parameter space that gives the *exact* lattice widths ($\kappa = 1$).

3.1 Description of the Method

In the following, we give an informal description of the method of Eisenbrand and Shmonin. In the defining equation (2.3) for the width of the polytope along any given direction $\mathbf{c}$, it is clear that the minimum and maximum are attained at (different) vertices of $P_{\mathbf{s},\mathbf{t}}$. The idea of the algorithm is then to consider all pairs of basic solutions that correspond to vertices $\mathbf{v}(\mathbf{s}, \mathbf{t})$ of $P_{\mathbf{s},\mathbf{t}}$, to compute all candidate integer directions for a given pair of such vertices and then to compute the minimum width over all candidate integer directions found. For any given basic solution $\mathbf{v}(\mathbf{s}, \mathbf{t})$, the (rational) directions for which this vertex is minimal form the inner normal cone C^* of the vertex. Suppose we have a pair of basic solutions, $\mathbf{v}_1(\mathbf{s}, \mathbf{t})$ attaining the minimum and $\mathbf{v}_2(\mathbf{s}, \mathbf{t})$ attaining the maximum; thus the set of (rational) directions for this pair of vertices is the "truncated cone" $C_{1,2} = (C_1^* \cap -C_2^*) \setminus \{\mathbf{0}\}$. A sufficient set of candidate *integer* directions are the vertices of the integer hull of $C_{1,2}$; all other integer directions in $C_{1,2}$ are dominated. As Eisenbrand–Shmonin observed, it now follows from [13] that this set of candidates is of polynomial size (in fixed dimension). After computing all candidates $\mathbf{c}_i$, we construct the chambers where each of the widths is minimal, i.e.,

$$Q_i = \{\, (\mathbf{s}, \mathbf{t}) \in Q \mid \forall j : \operatorname{width}_{\mathbf{c}_i} P_{\mathbf{s},\mathbf{t}} \leq \operatorname{width}_{\mathbf{c}_j} P_{\mathbf{s},\mathbf{t}} \,\}.$$

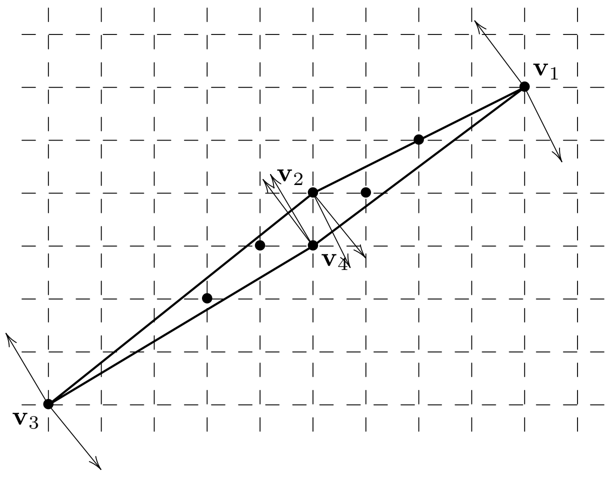

Figure 3.1: A polytope and its candidate width directions

Many of the Q_i will be empty or of lower dimension. Note that some of the Q_i will have nonempty intersection (along common facets). To obtain a *partition* of Q, Eisenbrand–Shmonin now use a lexicographic technique to replace the Q_i by partially open chambers $\tilde{Q}_i$.

3.2 Algorithmic Refinements

Depending on the parameter space Q, there can be basic solutions that are never primal feasible. Moreover, in the case of non-simple polytopes, several basic solutions correspond to the same parametric vertex. Instead of using all pairs of simplex bases, as proposed by Eisenbrand and Shmonin, we therefore consider all pairs of parametric vertices that become active on the parameter space Q.

Instead of Eisenbrand–Shmonin's lexicographic technique, we use the technique of [14], which yields a partition into partially open *full-dimensional* chambers $\tilde{Q}_i$.

In the following two sections, we discuss two implementation strategies for computing the integer hulls of the truncated cones $C_{1,2}$.

3.3 Using Generalized Basis Reduction

One way of computing the integer hull of a truncated cone $C_{1,2}$ is based on generalized basis reduction. We first describe how to compute the convex hull of a set given as an oracle for optimizing a linear objective function over the set. Then we explain how to optimize a linear objective function over the integer points of a polyhedron, using an integer feasibility test implemented with generalized basis reduction. Applying the first with the second as *optimization oracle* yields a method for computing the requested integer hull.

3.3.1 Computing the convex hull based on an optimization oracle

The algorithm described below is an application of [15, Remark 2.5]; see also [16, 17, 18, 19]. Let $\{\mathbf{r}_j\}$ be the smallest integer representatives of the extremal rays of the truncated cone $C_{1,2}$ (Figure 3.2). The algorithm starts out from the polyhedron $\operatorname{conv}\{\mathbf{r}_j\} + \operatorname{pos}\{\mathbf{r}_j\}$, which is certainly contained in the convex hull of $S = C_{1,2} \cap \mathbb{Z}^m$. We then take one of its facets and use the optimization oracle to compute a point in S that maximizes the outer normal of this facet. If a new point is found, it is added to the set of points and a new (larger) convex hull is computed. If not, the facet is guaranteed to be a facet of the convex hull of S.

3.3.2 Minimizing a linear function over the integer points of a polyhedron

For this task, using a general-purpose IP solver such as *CPLEX* [20] is a possibility, but it may easily run into numerical difficulties for the problems at hand. Instead we use *binary search* based on an exact integer feasibility test (oracle). Such an integer feasibility test is possible using the technique of [21], based on *generalized basis reduction*. The technique basically looks for a "short vector" $\mathbf{c}$ in the lattice $\mathbb{Z}^d$, where shortness is measured in terms of the width of the polytope P along that direction, and then branches on hyperplanes orthogonal to this direction. The computation of the reduced basis requires the solution of many linear programs, for which we use any of the solvers *GLPK* [22], *cdd* [23], *piplib* [24]. We remark that these solvers have very different characteristics.

3.4 Using Hilbert Bases

The *Hilbert basis* of a pointed cone C is the minimal set of points $\mathbf{b}_i \in C \cap \mathbb{Z}^d$ such that every integer point $\mathbf{x} \in C \cap \mathbb{Z}^d$ can be written as a non-negative *integer* combination of the $\mathbf{b}_i$. Thus it is a superset of the set of vertices of the integer hull of $C \setminus \{\mathbf{0}\}$ (Figure 3.2). Though no polynomiality results are available for Hilbert bases (they are typically exponentially large), efficient software implementations are available, such as the *zsolve* library from *4ti2* [25], which implements the technique of [26]. Hence,

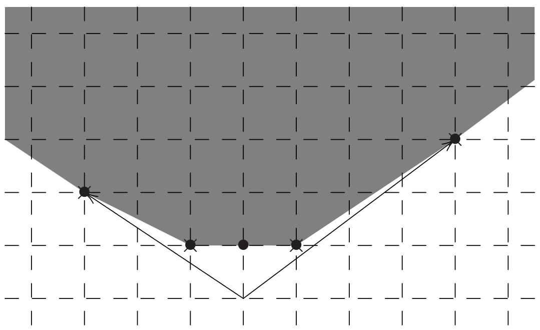

Figure 3.2: The Hilbert basis and the integer hull of a truncated cone.

an alternative practical way of computing the integer hull is to first compute the *Hilbert basis* of C and to remove the points that are not vertices of the integer hull of $C \setminus \{\mathbf{0}\}$. The latter can be done by linear programming.

4 Algorithmic Refinements

4.1 Stopping early

The only remaining problem is that the "$\odot$" operation in (2.7) is fairly expensive. The key step is the computation of the *Hadamard product* ($\star$) of the two generating functions, defined by

$$\sum_{\mathbf{s}} a_{\mathbf{s}}\, \mathbf{x}^{\mathbf{s}} \star \sum_{\mathbf{s}} b_{\mathbf{s}}\, \mathbf{x}^{\mathbf{s}} = \sum_{\mathbf{s}} a_{\mathbf{s}} b_{\mathbf{s}}\, \mathbf{x}^{\mathbf{s}}.$$

Computing the Hadamard product has a time complexity which, while polynomial if the dimension (in this case the maximum of the s_i in (1.3)) is fixed, is exponential in this dimension. Furthermore, this dimension increases by $\max_i s_i - d > 0$ on each successive application of the Hadamard product. Now, the total number of Hadamard products we must compute is bounded by a constant $\lfloor \omega(m) \rfloor$ and so the increase in dimension is also bounded by a constant, so the whole operation is still polynomial for fixed dimension. Nevertheless, we do not want to perform more Hadamard products than we have to. That is, we would like to be able to detect when we can stop performing these operations *before* reaching the upper bound $\lfloor \omega(m) \rfloor$.

Let $f'_0(\mathbf{x}, \mathbf{y}, z) = f'(\mathbf{x}, \mathbf{y}, z)$ and let $f'_k(\mathbf{x}, \mathbf{y}, z)$ be the result of applying the "set difference" in (2.7) k times. Denote the corresponding sets by T'_0 and T'_k. We want to find the smallest k such that $f''(\mathbf{x}, \mathbf{y}, z) = f'_k(\mathbf{x}, \mathbf{y}, z)$. Any further application of the set difference operation will not change this rational function, but it *will* typically produce a different (more complex) representation. To check whether the current k is sufficient, we are going to see whether any element of T'_k still appears in a shifted copy of T'_0, with shift greater than or equal to $k+1$. That is, we want to decide if the following set is empty:

$$\bigcup_{l=k+1}^{\infty} \Big(T'_k \cap (l\mathbf{e}_{n+d+1} + T') \Big).$$

We compute the corresponding generating function

$$\begin{aligned} h(\mathbf{x}, \mathbf{y}, z) &= \sum_{l=k+1}^{\infty} f'_k(\mathbf{x}, \mathbf{y}, z) \star z^l\, f'(\mathbf{x}, \mathbf{y}, z) \\ &= f'_k(\mathbf{x}, \mathbf{y}, z) \star \left(\sum_{l=k+1}^{\infty} z^l\, f'(\mathbf{x}, \mathbf{y}, z) \right) \\ &= f'_k(\mathbf{x}, \mathbf{y}, z) \star \frac{z^{k+1}\, f'(\mathbf{x}, \mathbf{y}, z)}{1-z}. \end{aligned}$$

The current k is sufficient if and only if $h(\mathbf{x}, \mathbf{y}, z)$ is identically zero. It suffices to check whether $h(\mathbf{1}, \mathbf{1}, 1) = 0$. We note that some care needs to be taken here since we allow the polyhedron P to be unbounded.

Lemma 4.1 *Fix k and s. Given a rational generating function $g(S; \boldsymbol{\xi})$ of the form (1.3) with $s_i \le s$ and such that $S \subset Q$, with $Q \subset \mathbb{Q}^k$ a pointed (but possibly unbounded) polyhedron. Then there exists a polynomial time algorithm that determines whether $g(S; \mathbf{1})$ diverges and computes the value of $g(S; \mathbf{1})$ if it does not.*

We omit the proof.

Note that testing whether we can stop is more expensive than applying the next iteration (since we have an extra $(1-z)$ factor in the denominator of one of the operands). However, we may save many iterations by stopping early and we will not needlessly replace a given representation of $f''(\mathbf{x}, \mathbf{y}, z)$ by a more complex representation (with more factors in the denominator).

4.2 Small gaps in low dimensions

There is a possibility for improving the bound of the small-gaps theorem (Theorem 2.5), at least for special cases or low dimensions. We present a first result for two-dimensional polytopes, where we can prove that the gaps in the lattice width direction will actually always be one. Note that the $\omega(2)$ bound is too coarse to reach this conclusion as $\omega(2) > 2$.

Table 1: Computational results

		Problem				
		ex1	*woods_2.1.7*	*pugh*	*param. pugh*	*scarf1*
Parameter variables **s**	n	0	1	0	1	2
Variables **t**	d	0	0	0	0	0
Existentially quantified variables **u**	m	2	2	2	2	2
Inequalities		4	4	4	4	5
Parametric Vertices		4	4	4	4	6
Width directions		7	3	8	8	6
Distinct width directions		4	2	7	7	4
Chambers		1	2	1	6	2
LPs solved in generalized basis reduction		8		49	43	4
Without exploiting small gaps in dimension 2						
Computation time (CPU seconds)		0.11	29.2	0.09	797	126
Exploiting small gaps in dimension 2						
Computation time (CPU seconds)		0.08	2.7		18.0	1.1

Lemma 4.2 *For any rational polygon, there is a lattice width direction whose gaps are always one.*

We omit the proof.

5 Computational Experiments

We show the results of preliminary computational experiments with the implementation in Table 1. We have run the implementation on several test problems, using the generalized basis reduction technique described in subsection 3.3. *ex1* is the example from Figure 2.1; *woods_2.1.7* and *scarf1* are examples that come with the library *barvinok* [6]; *pugh* is the "Omega nightmare" from [27] and *param. pugh* is a parametric version of this nightmare. We also made initial experiments using the Hilbert basis technique, but the results were disappointing; employing additional truncation techniques might help.

The table also shows the effect of using tighter gap-length estimates. If we use the improved estimate of Lemma 4.2, the running time and complexity of the resulting formula is reduced dramatically.

In future experiments, we will try out larger problems and also new algorithmic refinements.

References

[1] A. I. Barvinok, "Computing the Ehrhart polynomial of a convex lattice polytope," *Discrete Comput. Geom.*, vol. 12, pp. 35–48, 1994.

[2] A. I. Barvinok and K. Woods, "Short rational generating functions for lattice point problems," *Journal of the AMS*, vol. 16, no. 4, pp. 957–979, 2003.

[3] B. Sturmfels and S. Hoşten, "Computing the integer programming gap," *Combinatorica*, vol. 27, pp. 367–382, 2007.

[4] J. A. De Loera, R. Hemmecke, and M. Köppe, "Pareto optima of multicriteria integer linear programs," eprint arXiv:0707.1362 [math.OC], 2007, to appear in: INFORMS Journal on Computing.

[5] M. Köppe, M. Queyranne, and C. Ryan, "Algebraic techniques for bilevel optimization and game theory," 2008, manuscript.

[6] S. Verdoolaege, "`barvinok`, version 0.26," Available from URL http://freshmeat.net/projects/barvinok/, 2008.

[7] S. Verdoolaege and K. M. Woods, "Counting with rational generating functions," *J. Symb. Comput.*, vol. 43, no. 2, pp. 75–91, 2008.

[8] J. C. Lagarias, H. W. Lenstra, Jr., and C.-P. Schnorr, "Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice," *Combinatorica*, vol. 10, no. 4, pp. 333–348, 1990.

[9] A. Barvinok, *A Course in Convexity*, ser. Graduate Studies in Mathematics. Providence, RI: American Mathematical Society, 2002, vol. 54.

[10] W. Banaszczyk, A. E. Litvak, A. Pajor, and S. J. Szarek, "The flatness theorem for nonsymmetric convex bodies via the local theory of Banach spaces," *Mathematics of Operations Research*, vol. 24, no. 3, pp. 728–750, Aug. 1999.

[11] R. Kannan, "Lattice translates of a polytope and the Frobenius problem," *Combinatorica*, vol. 12, no. 2, pp. 161–177, 1992.

[12] F. Eisenbrand and G. Shmonin, "Parametric integer programming in fixed dimension," 2008, http://arXiv.org/abs/0801.4336.

[13] W. J. Cook, M. E. Hartmann, R. Kannan, and C. McDiarmid, "On integer points in polyhedra," *Combinatorica*, vol. 12, no. 1, pp. 27–37, 1992.

[14] M. Köppe and S. Verdoolaege, "Computing parametric rational generating functions with a primal Barvinok algorithm," *The Electronic Journal of Combinatorics*, vol. 15, pp. 1–19, 2008, #R16.

[15] W. Cook, M. Hartmann, R. Kannan, and C. McDiarmid, "On integer points in polyhedra," *Combinatorica*, vol. 12, no. 1, pp. 27–37, 1992.

[16] J. Edmonds, L. Lovász, and W. R. Pulleyblank, "Brick decompositions and the matching rank of graphs," *Combinatorica*, vol. 2, no. 3, pp. 247–274, 1982.

[17] F. Eisenbrand, "Gomory-Chvátal cutting planes and the elementary closure of polyhedra," Ph.D. dissertation, Universität des Saarlandes, Jul. 2000.

[18] M. E. Hartmann, "Cutting planes and the complexity of the integer hull," Ph.D. dissertation, Ithaca, NY, USA, 1989.

[19] P. Huggins, "*iB4e*: A software framework for parametrizing specialized LP problems," in *ICMS 2006, Proceedings of the Second International Congress on Mathematical Software*, ser. Lecture Notes in Computer Science, A. Iglesias and N. Takayama, Eds., vol. 4151. Springer, 2006, pp. 245–247.

[20] ILOG, "CPLEX," 1997–2007, http://www.ilog.com/products/cplex/. [Online]. Available: http://www.ilog.com/products/cplex/

[21] W. Cook, T. Rutherford, H. E. Scarf, and D. F. Shallcross, "An implementation of the generalized basis reduction algorithm for integer programming," *ORSA Journal on Computing*, vol. 5, no. 2, 1993.

[22] A. Makhorin, "GNU linear programming kit, reference manual, version 4.11," Jul. 2006.

[23] K. Fukuda, "cdd.c: C-implementation of the double description method for computing all vertices and extremal rays of a convex polyhedron given by a system of linear inequalities," Department of Mathematics, Swiss Federal Institute of Technology, Lausanne, Switzerland, Tech. Rep., 1993, program available from http://www.ifor.math.ethz.ch/~fukuda/fukuda.html.

[24] P. Feautrier, "Solving systems of affine (in)equalities: PIP's user's guide," 2006.

[25] R. Hemmecke, R. Hemmecke, M. Köppe, P. Malkin, and M. Walter, "4ti2 – a software package for algebraic, geometric and combinatorial problems on linear spaces," Available at www.4ti2.de.

[26] R. Hemmecke, "On the computation of Hilbert bases of cones," in *Mathematical Software, ICMS 2002*, A. M. Cohen, X.-S. Gao, and N. Takayama, Eds. World Scientific, 2002.

[27] W. Pugh, "A practical algorithm for exact array dependence analysis," *Communications of the ACM*, vol. 35, no. 8, pp. 102–114, 1992.