Short rational generating functions and their applications to integer programming

Kevin Woods
Department of Mathematics
University of California, Berkeley
Berkeley, CA 94720-3840
kwoods@math.berkeley.edu

Ruriko Yoshida
Department of Mathematics
Duke University
Durham, NC 27708-0320
ruriko@math.duke.edu

Abstract

In this paper, we provide an overview of rational generating function tools for encoding integer points inside a rational polyhedron, and we examine their applications to solving integer programming problems.

1 Introduction

In the 1980's, H. Lenstra discovered an algorithm to *detect* integer points in a rational polyhedron using the LLL-algorithm [18, 21]. As a consequence, Lenstra showed that integer programming problems with a fixed number of variables can be solved in time polynomial in the input size. A later algorithm of similar structure, by Lovász and Scarf [22], was implemented by Cook, et al. [10]. In addition, Aardal and collaborators [1, 2, 3] have written fairly effective modifications of the LLL-procedure for testing integer feasibility. In the 1990's, based on work by the geometers Brion, Khovanski, Lawrence, and Pukhlikov, Barvinok discovered an algorithm to *count* integer points inside rational polytopes, and this algorithm also runs in polynomial time if we fix the dimension [6, 7]. The idea of the algorithm is to encode all the integer points inside a rational polyhedron into a *rational generating function*. In particular, if $P \subset \mathbb{R}^d$ is a rational polyhedron, define the generating function

$$f(P;\mathbf{x}) = \sum_{s\in P\cap\mathbb{Z}^d} \mathbf{x}^s,$$

where $\mathbf{x}^s$ denotes $x_1^{s_1}\cdots x_d^{s_d}$ with $s=(s_1,\ldots,s_d)$. After computing $f(P;\mathbf{x})$ as a rational function, we evaluate $|P\cap\mathbb{Z}^d| = f(P;1,\ldots,1)$.

Example 1. *Suppose P is the one-dimensional polytope $[0,N]$. Then $f(P;x)=1+x+x^2+\cdots+x^N$, $f(P;x)$ can be represented by the rational function $\frac{1-x^{N+1}}{1-x}$, and $f(P;1)=N+1$, the number of integer points in P.*

Note that the rational function representation of $f(P;\mathbf{x})$ in Example 1 is "short." In general, the number of terms in the representation will be bounded by a polynomial in the input size. Also note that 1 is a zero of the denominator of the rational function, so some analytic technique must be used to evaluate $f(P;1)$. In this particular case, we could take the limit as x approaches 1 and apply l'Hospital's rule. In general, we must use more complicated residue calculus [6].

Shortly after Barvinok introduced his algorithm, Dyer and Kannan [17] showed that a particular step of his counting algorithm, which originally relied on Lenstra's result, could be replaced by a short-vector computation using the LLL-algorithm. This result gives a new proof that integer programming problems with a fixed number of variables can be solved in polynomial time: using binary search, one can turn

Barvinok's counting oracle into an algorithm which solves integer programming problems. This algorithm was proposed by Barvinok in [7]. We call this integer programming algorithm the *BBS algorithm*.

In this report, we will provide a brief overview of how to encode the integer points inside a given polyhedron into a short rational generating function (Section 2), and then we will survey applications to solving integer programming problems (Section 3).

2 Computing short rational generating functions

In 1993, Barvinok [6] gave an algorithm that, given a rational polyhedron $P \subset \mathbb{R}^d$, computes $f(P;\mathbf{x})$ as a rational function in polynomial time (when the dimension of the polyhedron is fixed). In this section, we outline the steps of Barvinok's algorithm. For a more lengthy and detailed exposition, see [7].

Let $\{c_1, \ldots, c_d\}$ be a basis for the lattice $\mathbb{Z}^d$, and let $\beta_1, \ldots, \beta_d \in \mathbb{Q}$ be given. We define the *rational unimodular cone* $K = \{x \in \mathbb{R}^d : \ c_i \cdot x \leq \beta_i, \ \text{for all } i\}$, where $a \cdot b$ is the standard dot product. Computing the generating function of a unimodular cone is "easy." In fact, if $\{u_1, \ldots, u_d\}$ is the (negative) dual basis of $\mathbb{Z}^d$ such that

$$u_i \cdot c_j = \begin{cases} -1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases},$$

and if $v = -\sum_{i=1}^d \lfloor \beta_i \rfloor u_i$, then (by Lemma 4.1 of [7])

$$f(K;\mathbf{x}) = \frac{\mathbf{x}^v}{(1-\mathbf{x}^{u_1})\cdots(1-\mathbf{x}^{u_d})}. \tag{1}$$

Example 2. *Suppose we have a rational unimodular cone $K \subset \mathbb{R}^2$, such that*

$$K = \{(x_1, \ x_2) \in \mathbb{R}^2 : x_1 \leq 4, \ 2x_1 + x_2 \leq 10\}.$$

In our example, $c_1 = (1, \ 0)$ and $c_2 = (2, \ 1)$ form a basis of $\mathbb{Z}^2$. Then, we have that $u_1 = (-1, \ 2)$, $u_2 = (0, \ -1)$, and $v = -[4(-1, \ 2) + 10(0, \ -1)] = (4, \ 2)$. Therefore

$$f(K; x_1, x_2) = \frac{x_1^4 x_2^2}{(1 - x_1^{-1} x_2{}^2)(1 - x_2^{-1})}.$$

The idea of Barvinok's algorithm, then, is to reduce the case of computing $f(P;\mathbf{x})$ to the case of computing the generating functions for a collection of unimodular cones. We do this in three steps: first reduce to the case of general rational cones, then to the case of simplicial cones (rational cones with exactly d extreme rays), and then finally to unimodular cones. If v is a vertex P, then define the *supporting cone*, $\text{cone}(P, v)$, of P at v, as follows. Suppose that P is defined by $P = \{x \in \mathbb{R}^d : \ c_i \cdot x \leq \beta_i, \ \text{for } i = 1, \ldots, m\}$, for some $c_i \in \mathbb{Q}^d$, $\beta_i \in \mathbb{Q}$. For a vertex, v, of P, let $I_v = \{i : \ c_i \cdot v = \beta_i\}$, and define

$$\text{cone}(P, v) = \{x \in \mathbb{R}^d : \ c_i \cdot x \leq \beta_i, \ \text{for } i \in I_v\}.$$

Then the following theorem from [9] allows us to compute $f(P, \mathbf{x})$ by computing the generating functions of the cones $\text{cone}(P, v)$, where v is a vertex of P.

Theorem 3. *(Brion's Theorem) Let P be a rational polyhedron. Then*

$$f(P;\mathbf{x}) = \sum_v f\big(\text{cone}(P, v); \mathbf{x}\big),$$

where the sum runs over all vertices v of P.

Example 4. *Let P be the one-dimensional polytope $[0, N]$. Then $v_0 = 0$ and $v_1 = N$ are the vertices of P, $\operatorname{cone}(P, v_0) = [0, \infty)$ and $\operatorname{cone}(P, v_1) = (-\infty, N]$, and so*

$$f(P; x) = f(\operatorname{cone}(P, v_0); x) + f(\operatorname{cone}(P, v_1); x) = \frac{1}{1-x} + \frac{x^N}{1-x^{-1}} = \frac{1-x^{N+1}}{1-x}.$$

Note that, since the dimension d is fixed, we may compute the vertices of P in polynomial time. Next, by triangulating these cones, we reduce to the case of simplicial cones. There are efficient algorithms, when the dimension is fixed, to perform triangulations (see [5, 20] for details). Finally, let K be a simplicial cone. The essential contribution of Barvinok [6] was to show that we can decompose K into a *signed* collection of unimodular cones. To be precise, given a set $A \subset \mathbb{R}^d$, define the indicator function, $[A] : \mathbb{R}^d \to \mathbb{R}$, of A by

$$[A](x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}.$$

Then we have the following theorem from [6].

Theorem 5. *Fix d. There is a polynomial time algorithm which, given a rational simplicial cone $K \subset \mathbb{R}^d$, computes rational unimodular cones K_i and signs $\epsilon_i \in \{-1, 1\}$ such that*

$$[K] = \sum_i \epsilon_i [K_i].$$

Therefore, we have that

$$f(K, \mathbf{x}) = \sum_i \epsilon_i f(K_i, \mathbf{x}).$$

Example 6. *Let $K = \operatorname{co}\big((1,0),(1,N)\big)$ (that is, the cone generated by $(1,0)$ and $(1,N)$). Then $K_1 = \operatorname{co}\big((1,0),(0,1)\big)$, $K_2 = \operatorname{co}\big((1,N),(0,1)\big)$, and $K_3 = \operatorname{co}\big((1,N)\big)$ are unimodular, with $[K] = [K_1] - [K_2] + [K_3]$. Therefore*

$$f(K; x, y) = \frac{1}{(1-x)(1-y)} - \frac{1}{(1-xy^N)(1-y)} + \frac{1}{1-xy^N}.$$

Using Theorem 3, triangulation, Theorem 5, and (1), we may now compute the generating function for any rational polyhedron, as the following theorem states.

Theorem 7 (Theorem 4.4 in [7])**.** *Assume d, the dimension, is fixed. Given a rational polyhedron $P \subset \mathbb{R}^d$, the generating function $f(P; \mathbf{x})$ can be computed in polynomial time in the form*

$$f(P; \mathbf{x}) = \sum_{i \in I} \epsilon_i \frac{\mathbf{x}^{u_i}}{\prod_{j=1}^{d} (1 - \mathbf{x}^{v_{ij}})}, \tag{2}$$

where I is a polynomial-size indexing set, and where $\epsilon_i \in \{1, -1\}$ and $u_i, v_{ij} \in \mathbb{Z}^d$ for all i and j.

Furthermore, we can use this generating function to count the number of integer points in a rational polytope, by computing $f(P; 1, 1, \ldots, 1)$. Note that, while $f(P; \mathbf{x})$ itself is analytic at $(1, 1, \ldots, 1)$, each of the fractions in the sum has a pole there. Thus, we can not directly substitute 1 for each variable, but we can compute the value (in polynomial time for fixed dimension) via residue calculus, as in [6].

3 Applications to integer programming

Throughout this section, we will assume that the number of variables d is fixed. Suppose we have the following integer programming problem: given $A \in \mathbb{Z}^{m \times d}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^d$,

$$\text{maximize } c \cdot x \text{ subject to } Ax \le b,\ x \in \mathbb{Z}^d. \tag{3}$$

There are several algorithms to solve the integer programming problem (3) using short rational functions [13, 14, 15]. In this section, we will provide outlines of two of these algorithms: the *BBS algorithm* and the *digging algorithm*. These are currently implemented in the second release of the computer software `LattE` (see [12, 13, 16]).

3.1 Barvinok's binary search algorithm

We start with the most straightforward integer programming algorithm. It is an immediate consequence of the counting algorithm via short rational functions, with no extra tools needed: using binary search, one can turn *any* feasibility or counting oracle into an algorithm that solves Problem (3). This idea was proposed in [7]:

Algorithm 8. *(BBS algorithm)*
Input: $A \in \mathbb{Z}^{m\times d}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^d$.
Output: *The optimal value, O, of* $\max\{c\cdot x : Ax \le b,\ x \in \mathbb{Z}^d\}$.

1. *Let $P = \{x \in \mathbb{R}^d :\ Ax \le b\}$ be the feasible region for the linear program. Initialize M to be $\max\{c\cdot x :\ x \in P\}$, the maximum of the linear program, and m to be $\min\{c \cdot x :\ x \in P\}$. Then $m \le O \le M$ (assuming a feasible integer solution exists).*
2. *Use Barvinok's algorithm to count $q = \left|P \cap \mathbb{Z}^d\right|$. If $q = 0$, then there is no feasible integer solution. If q is nonzero, then perform the following loop.*
3. *While $M > m$ do (at each iteration, we will know that $m \le O \le M$)*
   Set $N = \lceil \frac{M+m}{2} \rceil$.
   Using Barvinok's algorithm compute $q = \left|P \cap \{x \in \mathbb{R}^d :\ N \le c\cdot x \le M\} \cap \mathbb{Z}^d\right|$.
   If $q > 0$, then $N \le O \le M$ and we repeat the loop with $m := N$ and $M := M$.
   If $q = 0$, then $m \le O < N$ and we repeat the loop with $m := m$ and $M := N - 1$.
4. *Return M as the optimal value.*

3.2 Digging algorithm

An alternative integer programming algorithm, which also uses rational generating functions, is the *digging algorithm.* This algorithm is an extension of a heuristic proposed by Lasserre [19], and it begins at the highest possible value for the optimum of the integer program, checks to see if there is a feasible solution giving that value, and if not continually *digs* to check the next highest possible value.

For the integer programming problem (3), let $P = \{x \in \mathbb{R}^d :\ Ax \le b\}$ be the feasible region for the *linear* program. Given the generating function $f(P; \mathbf{x})$ as in (2), perform the substitution $x_k = t^{c_k}$, so that

$$f(P; t^{c_1}, \cdots ; t^{c_d}) = \sum_{\alpha \in P\cap\mathbb{Z}^d} t^{c\cdot\alpha} = \sum_{i\in I} \epsilon_i \frac{t^{c\cdot u_i}}{\prod_{j=1}^{d}(1 - t^{c\cdot v_{ij}})}. \tag{4}$$

For simplicity, we assume that $c \cdot v_{ij} \neq 0$, for any i, j. Furthermore, we may assume that $c \cdot v_{ij} < 0$ by performing the operation $1/(1 - t^v) = -t^{-v}/(1 - t^{-v})$, as necessary.

The optimum value of the integer program is the degree of $f(P; t^{c_1}, \cdots , t^{c_d})$, which we will compute by examining the Laurent series expressions of each of the terms in the sum, which are of the form

$$\epsilon_i t^{c\cdot u_i} \prod_{j=1}^{d} (1 + t^{c\cdot v_{ij}} + t^{2c\cdot v_{ij}} + t^{3c\cdot v_{ij}} + \cdots). \tag{5}$$

Algorithm 9. *(Digging algorithm)*
Input: $A \in \mathbb{Z}^{m\times d}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^d$.
Output: *The optimal value of* $\max\{c\cdot x : Ax \le b,\ x \in \mathbb{Z}^d\}$.

1. *Let $P = \{x \in \mathbb{R}^d :\ Ax \le b\}$. Using Theorem 7 and substitution, compute the rational function expression in the equation (4). Apply the appropriate algebraic identities so that $c \cdot v_{ij} < 0$ for all i, j.*
2. *If $M = \max_i c \cdot u_i$, then the degree of $f(P; t^{c_1}, \cdots , t^{c_d})$ is at most M (since all of the $c \cdot v_{ij}$ are negative). Use the formula (5) to compute the coefficient of t^M in $f(P; t^{c_1}, \cdots , t^{c_d})$.*

3. *If the coefficient of t^M is nonzero, then M is the optimal value of the integer program and we are done. Otherwise, compute the coefficient of the next highest possible degree. Continue until a nonzero coefficient is found.*

Example 10. *Suppose we have the integer programming problem*

$$\textit{maximize}\ \ 100x + 90y \textit{ subject to } x + y \leq 100,\ x \leq 50,\ x, y \geq 0,\ x, y \in \mathbb{Z}^d.$$

Then the associated rational generating function for the feasible region of the integer program is

$$\frac{1}{(1-x_1)(1-x_2)} + \frac{x_1^{50}}{(1-x_1^{-1})(1-x_2)} + \frac{x_2^{100}}{(1-x_2^{-1})(1-x_1x_2^{-1})} + \frac{x_1^{50}x_2^{50}}{(1-x_2^{-1})(1-x_1^{-1}x_2)}.$$

We apply the monomial substitution in (4) *and apply the appropriate algebraic identities, and we get:*

$$\frac{t^{-190}}{(1-t^{-100})(1-t^{-90})} - \frac{t^{4910}}{(1-t^{-100})(1-t^{-90})} - \frac{t^{8990}}{(1-t^{-90})(1-t^{-10})} + \frac{t^{9500}}{(1-t^{-90})(1-t^{-10})}.$$

Then, using Equation (5) *and the fact that the coefficient of t^{9500} is nonzero, we find that the integer programming optimum is* 9500.

3.3 Comparison of BBS and digging algorithms

How do these two algorithms compare? The Barvinok binary search algorithm is guaranteed to run in polynomial time if we fix d. However, it is usually slow in practice, because the Barvinok counting algorithm must be run a number of times. The digging algorithm, on the other hand, is not guaranteed to run in polynomial time with fixed d. Nevertheless, it is often quite efficient in practice, because the number of iterations in Step 3 of Algorithm 9 is usually low and Barvinok's algorithm is called only once (even if we vary cost functions, we do not have to re-run Barvinok's algorithm). This algorithm works well for small dimensions and polytopes with a small number of vertices. For example, the digging algorithm solved some hard knapsack problems which the mixed-integer programming solver `CPLEX` version 6.6. could not [14]. However, for d on the order of 30, this algorithm becomes quite slow. If the input polytope has a large number of vertices, the integer programming problem (3) can be solved by the following: First we find a vertex of the polytope which is an optimal solution for the linear relaxation of the problem (3) via linear programming. Then, we apply the digging calculation to the single tangent cone of the polytope at the vertex (see details in [14]).

References

[1] Aardal, K. and Lenstra, A.K. *Hard equality constrained integer knapsacks* Preliminary version in W.J. Cook and A.S. Schulz (eds.), Integer Programming and Combinatorial Optimization: 9th International IPCO Conference, Lecture Notes in Computer Science vol. 2337, Springer-Verlag, 2002, 350–366.

[2] Aardal, K. Weismantel, R., and Wolsey, L.A. *Non-standard approaches to integer programming.* Workshop on Discrete Optimization, DO'99 (Piscataway, NJ). Discrete Appl. Math. 123, 2002, no. 1-3, 5–74.

[3] Aardal, K., Hurkens, C.A.J., and Lenstra, A.K. *Solving a linear diophantine equations with lower and upper bounds on the variables.* In R.E Bixby, E.A Boyd, R.Z. Rios-Mercado (eds) "Integer Programming and Combinatorial Optimization", 6th International IPCO conference. Lecture notes in Computer Science 1412 Springer Verlag, 1998, 229-242.

[4] Aardal, K., Weismantel, R., and Wolsey, L. *Non-standard approaches to integer programming.* Discrete Applied Mathematics 123, 2002, 5–74

[5] Aurenhammer, F. and Klein, R. *Voronoi diagrams.* In *Handbook of computational geometry*. (Ed. J.-R. Sack and J. Urrutia). Amsterdam, Netherlands: North-Holland, 2000, 201–290.

[6] Barvinok, A. *Polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed.* Math of Operations Research 19, 1994, 769–779.

[7] Barvinok, A. and Pommersheim, J. *An algorithmic theory of lattice points in polyhedra. New Perspectives in Algebraic Combinatorics* (Berkeley, CA, 1996-1997), 91–147, Math. Sci. Res. Inst. Publ. 38, Cambridge Univ. Press, Cambridge, 1999.

[8] Barvinok, A. and Woods, K. *Short rational generating functions for lattice point problems.* J. Amer. Math. Soc. 16, 2003, 957-979.

[9] Brion, M. *Points entiers dans les polyèdres convexes.* Ann. Sci. École Norm. Sup. (4), 21, 1988, 653–663.

[10] Cook, W., Rutherford, T., Scarf, H.E., and Shallcross, D. *An implementation of the generalized basis reduction algorithm for integer programming.* ORSA Journal of Computing, 5, 1993, 206–212.

[11] Cornuéjols, G., Urbaniak, R., Weismantel, R., and Wolsey, L.A. *Decomposition of integer programs and of generating sets.* R. E. Burkard, G. J. Woeginger, eds., *Algorithms*–ESA 97. Lecture Notes in Computer Science 1284, Springer-Verlag, 1997, 92–103.

[12] De Loera, J.A, Hemmecke, R., Tauzer, J., and Yoshida, R. *Effective lattice point counting in rational convex polytopes.* The Journal of Symbolic Computation, 38, 2004, no. 4, p1273 – 1302

[13] De Loera, J.A, Haws, D., Hemmecke, R., Huggins, P., Sturmfels, B., and Yoshida, R. *Short rational functions for toric algebra and applications.* The Journal of Symbolic Computation, 38, 2004, no. 2, p959–973.

[14] De Loera, J.A, Haws, D., Hemmecke, R., Huggins, P., and Yoshida, R. *A computational study of integer programming algorithms based on Barvinok's rational functions.* To appear in the Journal of Discrete Optimization.

[15] De Loera, J.A, Haws, D., Hemmecke, R., Huggins, P., and Yoshida, R. *Three kinds of integer programming algorithms based on Barvinok's rational functions.* Integer Programming and Combinatorial Optimization: 10th International IPCO Conference, Springer, (D. Bienstock and G. Nemhauser eds.) 2004, 244 – 255.

[16] De Loera, J.A., Haws, D., Hemmecke, R., Huggins, P., Tauzer, J., and Yoshida, R. *A user's guide for* `LattE` *v1.1.* 2003, software package `LattE` is available at `http://www.math.ucdavis.edu/~latte/`

[17] Dyer, M. and Kannan, R. *On Barvinok's algorithm for counting lattice points in fixed dimension.* Math of Operations Research 22, 1997, 545– 549.

[18] Grötschel, M., Lovász, L., and Schrijver, A. *Geometric algorithms and combinatorial optimization.* Second edition. Algorithms and Combinatorics, 2, Springer-Verlag, Berlin, 1993.

[19] Lasserre, J.B. *Integer programming, Barvinok's counting algorithm and Gomory relaxations.* Operations Research Letters, 32, 2003, 133–137.

[20] Lee, C.W. *Subdivisions and triangulations of polytopes.* In *Handbook of Discrete and Computational Geometry,* 271-290, (Goodman J.E. and O'Rourke J. eds.), CRC Press, New York, 1997.

[21] Lenstra, H.W. *Integer programming with a fixed number of variables.* Mathematics of Operations Research, 8, 1983, 538–548.

[22] Lovász, L. and Scarf, H.E. *The generalized basis reduction algorithm.* Math. of Operations Research, 17, 1992, 751–764.

[23] Schrijver, A. *Theory of linear and integer programming.* Wiley-Interscience, 1986.

[24] Thomas, R. *Algebraic methods in integer programming.* Encyclopedia of Optimization (eds: C. Floudas and P. Pardalos), Kluwer Academic Publishers, Dordrecht, 2001.