Fast integer programming in fixed dimension

Friedrich Eisenbrand

MPI–I–2003–NWG2-002 February 2003

Author's Address

Friedrich Eisenbrand
Max-Planck-Institut für Informatik
Im Stadtwald
66123 Saarbrücken, Germany
eisen@mpi-sb.mpg.de

Abstract

It is shown that the optimum of an integer program in fixed dimension with a fixed number of constraints can be computed with $O(s)$ basic arithmetic operations, where s is the binary encoding length of the input. This improves on the quadratic running time of previous algorithms, which are based on Lenstra's algorithm and binary search.
It follows that an integer program in fixed dimension, which is defined by m constraints, each of binary encoding length at most s, can be solved with an expected number of $O(m+\log(m)\,s)$ arithmetic operations using Clarkson's random sampling algorithm.

1 Introduction

An *integer program* is a problem of the following kind. Given an integral matrix $A \in \mathbb{Z}^{m \times n}$ and integral vectors $b \in \mathbb{Z}^m$, $d \in \mathbb{Z}^n$, determine

$$\max\{d^T x \mid Ax \leqslant b, x \in \mathbb{Z}^n\}. \tag{1}$$

It is well known [7] that integer programming is NP-complete. The situation changes, if the number of variables or the *dimension* is fixed. For this case, Lenstra [16] showed that (1) can be solved in polynomial time. Lenstra's algorithm does not solve the integer programming problem directly. Instead, it is an algorithm for the *integer feasibility problem*. Here, the task is to find an integer point which satisfies all the constraints, or to assure that $Ax \leqslant b$ is integer infeasible. If $Ax \leqslant b$ consists of m constraints, each of binary encoding length $O(s)$, then Lenstra's algorithm requires $O(m+s)$ arithmetic operations on rational numbers of size $O(s)$. The actual integer programming problem (1) can then be solved via *binary search*. It is known [19, p. 239] that, if there exists an optimal solution, then there exists one with binary encoding length $O(s)$. Consequently, the integer programming problem can be solved with $O(ms+s^2)$ arithmetic operations on $O(s)$-bit numbers. Lenstra's algorithm was subsequently improved [11, 1] by reducing the dependence of the complexity on the dimension n. However, these improvements do not affect the asymptotic complexity of the integer programming problem in fixed dimension. Unless explicitely stated, we from now-on assume that the dimension n is fixed.

Clarkson [2] presented a random sampling algorithm to reduce the dependence of the complexity on the number of constraints.* His result is the following. An integer program which is defined by m constraints can be solved with $O(m)$ basic operations and $O(\log m)$ calls to an algorithm which solves an integer program defined by a fixed size subset of the constraints, see also [8].

In light of these results, we are motivated to find a faster algorithm for the integer programming problem in fixed dimension with a fixed number of constraints. It is known [5] that the 2-dimensional integer programming problem with a fixed number of constraints can be solved in linear time. We generalize this to any fixed dimension.

Theorem 1. *An integer program of binary encoding length s in fixed dimension, which is defined by a fixed number of constraints, can be solved with $O(s)$ arithmetic operations on rational numbers of binary encoding length $O(s)$.*

With Clarkson's result, Theorem 1 implies that an integer program which is defined by m constraints, each of binary encoding length $O(s)$ can be solved with an expected number of $O(m+\log(m)\,s)$ arithmetic operations on rational numbers of binary encoding size $O(s)$. Our result was also motivated by the following fact. The greatest common divisor of two integers can be formulated as an integer program in fixed dimension with a fixed number of constraints, see, e.g., [14]. Our result matches the complexity of the integer programming approach to the gcd with the complexity of the Euclidean algorithm.

*Clarkson claims a complexity of $O(m+\log(m)\,s)$ because he mistakenly relied on algorithms from the literature [16, 11, 6] for the integer programming problem with a fixed number of constraints, which actually only solve the *integer feasibility problem*.

Outline of our method

As in Lenstra's algorithm, we make use of the lattice width concept. Let $K \subseteq \mathbb{R}^n$ be a full-dimensional convex body. The *width* of K *along a direction* $c \in \mathbb{R}^n$ is the quantity $w_c(K) = \max\{c^T x \mid x \in K\} - \min\{c^T x \mid x \in K\}$. The *width* of K, $w(K)$, is the minimum of its widths along nonzero integral vectors $c \in \mathbb{Z}^n \setminus \{0\}$. If K does not include any lattice points, then K must be "flat". This fact is known as Khinchin's *flatness theorem* (see [13]).

Theorem 2 (Flatness theorem). *There exists a constant f_n depending only on the dimension n, such that each full-dimensional convex body $K \subseteq \mathbb{R}^n$, containing no integer points has width at most f_n.*

This fact is exploited in Lenstra's algorithm [16, 9] for the integer feasibility problem as follows. If one has to decide, whether a full-dimensional polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\}$ is integer feasible or not, one computes an integral direction $c \in \mathbb{Z}^n \setminus \{0\}$ with $w(P) \leqslant w_c(P) \leqslant \gamma w(P)$ for some constant γ. If $w_c(P)$ is larger than γf_n, then P must contain integer points by the flatness theorem. Otherwise, an integer point of P must lie in one of the constant number of $(n-1)$-dimensional polyhedra

$$P \cap (c^T x = \delta), \text{ where } \delta \in \mathbb{Z} \cap [\min\{d^T x \mid x \in P\}, \max\{d^T x \mid x \in P\}].$$

In this way one can reduce the integer feasibility problem in dimension n to a constant number of integer feasibility problems in dimension $n-1$.

Our approach is to let the objective function slide into the polyhedron until the with of the truncated polyhedron $P \cap (d^T x \geqslant \pi)$ is sandwiched between $f_n + 1$ and $\gamma(f_n + 1)$. In this way, we assure that the optimum to the integer programming problem lies in the truncation $P \cap (d^T x \geqslant \pi)$. Thus we reduce the *integer programming problem* over a n-dimensional polyhedron to the *integer programming problem* over a constant number of $(n-1)$-dimensional polyhedra.

The problem of determining the correct parameter π is referred to as the *approximate parametric lattice width problem*. The 2-dimensional integer programming algorithm of Eisenbrand and Rote [4] makes already use of this concept. In this paper we generalize this approach to any dimension. We observe that a simplex, whose vertices partition into two groups, where each group agrees on the objective function value, allows a reduction of the approximate parametric lattice width problem to a parametric shortest vector problem. In two dimensions, such a partition is immediate as the objective function sweeps over the first intermediate vertex. In higher dimension one has to use triangulation. Furthermore, while the 2-dimensional approximate parametric shortest lattice width problem can be solved with the computation of a best approximation of a certain number, determined by the simplex, the approach here is to use the LLL-algorithm for lattice basis reduction.

1.1 Notation

A *polyhedron* P is a set of the form $P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\}$, for some matrix $A \in \mathbb{R}^{m \times n}$ and some vector $b \in \mathbb{R}^m$. The polyhedron is *rational* if both A and b can be chosen to be rational. If P is bounded, then P is called a *polytope*. The *dimension* of P is the dimension of the affine hull of P. The polyhedron $P \subseteq \mathbb{R}^n$ is *full-dimensional*, if its dimension is n. An inequality $c^T x \leqslant \delta$ defines a *face* $F = \{x \in P \mid c^T x = \delta\}$ of P, if $\delta \geqslant \max\{c^T x \mid x \in P\}$. If $F \neq \emptyset$ is a face of dimension 0, then F is called a *vertex* of P. A *simplex* is full-dimensional polytope $\Sigma \subseteq \mathbb{R}^n$ with $n+1$ vertices. We refer to [18] and [19] for further basics of polyhedral theory.

The *size* of an integer z is the number $\text{size}(z) = 1 + \lceil \log_2(|z|+1) \rceil$. The size of a rational is the sum of the sizes of its numerator and denominator. Likewise, the size of a matrix $A \in \mathbb{Z}^{m \times n}$ is the number of bits needed to encode A, i.e., $\text{size}(A) = \sum_{i,j} \text{size}(a_{i,j})$, see [19, p. 29]. If a polyhedron P is given as $P(A,b)$, then we denote $\text{size}(A) + \text{size}(b)$ by $\text{size}(P)$. A polytope can be represented by a set of constraints, as well as by the set of its vertices. In this paper we concentrate on polyhedra in fixed dimension with a fixed number of constraints. In this case, if a rational polytope is given by a set of constraints $Ax \leqslant b$ of size s, then the vertex representation $\text{conv}\{v_1, \ldots, v_k\}$ can be computed in constant time and the vertex representation has size $O(s)$. The same holds vice versa.

A rational *lattice* in $\mathbb{R}^n$ is a set of the form $\Lambda = \{Ax \mid x \in \mathbb{Z}^n\}$, where $A \in \mathbb{Q}^{n \times n}$ is a nonsingular matrix. This matrix is a *basis* of Λ and we say that Λ is generated by A and we also write $\Lambda(A)$ to denote a lattice generated by a matrix A. A *shortest vector* of Λ is a nonzero member $0 \neq v \in \Lambda$ of the lattice with minimal euclidean norm $\|v\|$. We denote the length of a shortest vector by $\text{SV}(\Lambda)$.

2 Proof of Theorem 1

Suppose we are given an integer program (1) in fixed dimension with a fixed number of constraints of binary encoding length s. We can assume without loss of generality that the polyhedron $P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\}$ is bounded, see, e.g. [19, p. 237]. Furthermore, with a constant number of calls to a linear programming algorithm in fixed dimension [17] and the Gaussian algorithm, one can compute a (possibly empty) subsystem $A'x \leqslant b'$ of $Ax \leqslant b$, such that A' has full row-rank and the affine hull of P is the set $\{x \in \mathbb{R}^n \mid A'x = b'\}$. Suppose that the rank of A' is k. By means of a unimodular transformation, which transforms A' to its *Hermite Normal Form*, see [19, p. 45] and [12, 3], the first k variables are set to a specific integer. Thus we can assume that P is full-dimensional. Again, by means of a unimodular transformation, which sends d^T to the vector $(\gcd(d(1), \ldots, d(n)), 0, \ldots, 0)$, we can also assume that the objective is to find an integer vector with maximal first component. These transformations can be done with $O(s)$ arithmetic operations on rational numbers of size $O(s)$. Thus we assume that an integer program has the following form.

Problem (IP). Let $m, n \in \mathbb{N}$ be fixed. Given an integral matrix $A \in \mathbb{Z}^{m \times n}$ and an integral vector $b \in \mathbb{Z}^m$ which define a full-dimensional polytope $P = \{x \in \mathbb{R}^n \mid Ax \leqslant b\}$, determine

$$\max\{x(1) \mid x \in P \cap \mathbb{Z}^n\}. \tag{2}$$

The *size* of an IP is the sum of the sizes of A and b.

Our main theorem is proved by induction on the dimension. We know that it holds for $n = 1, 2$ [5, 21, 10]. The induction step is by a series of reductions, for which we now give an overview.

(Step 1) We consider a reduction to, what we call, IP over two-layer simplices. A *two-layer polytope* has the property that its vertices can be partitioned into two sets V and W, such that the first components of all elements in one set agree.

(Step 2) We reduce IP over a two-layer simplex Σ to the problem of determining a parameter π, such that the width of the truncated simplex $\Sigma \cap (x(1) \geqslant \pi)$ is sandwiched between $f_n + 1$ and $(f_n + 1) \cdot \gamma$, where γ is a constant which depends on the dimension only. This problem is the *approximate parametric lattice width* problem.

(Step 3) We reduce the approximate parametric lattice width problem to an *approximate parametric shortest vector problem*. Here one is given a lattice basis A and parameters U and k. The task is to find a parameter p such that the length of the shortest vector of the lattice generated $A_{p,k}$ is sandwiched between U and $\gamma' U$, where γ' is a constant which depends on the dimension only. Here $A_{p,k}$ denotes the matrix, which evolves from A by scaling the first k rows with p.

(Step 4) We show that an approximate parametric shortest vector problem can be solved in linear time with a sequence of calls to the LLL-algorithm.

The linear complexity of the parametric shortest vector problem carries over to the integer programming problem with a fixed number of constraints, if we can ensure the following conditions for each reduction step.

(C-1) A problem of size s is reduced to a constant number of problems of size $O(s)$.

(C-2) The size of the rational numbers which are manipulated in the course of the reduction of a problem of size s, do not grow beyond $O(s)$.

At the end of each reduction step, we clarify that the conditions (C-1) and (C-2) are fulfilled.

2.1 Reduction to two-layer simplices

A *two-layer polytope* is a full-dimensional polytope, whose vertices can be partitioned into two sets V and W, such that the first components of the elements in V and W agree, i.e., for all $v_1, v_2 \in V$ one has $v_1(1) = v_2(1)$ and for all $w_1, w_2 \in W$ one has $w_1(1) = w_2(1)$.

Let P be a full dimensional polytope for which we want to solve IP. We reduce this problem to the disjunction of integer programs over two-layer simplices as follows.

First, compute the list of the first components $\alpha_1, \ldots, \alpha_\ell$ of the vertices of P in decreasing order. The optimal solution of IP over P is the largest optimal solution of IP over the two-layer polytopes

$$P_i = P \cap (x(1) \leqslant \alpha_i) \cap (x(1) \geqslant \alpha_{i+1}),\ i = 1, \ldots, \ell - 1. \qquad (3)$$

Each integer program over a two-layer polytope P_i in (3) can be reduced to a constant number of integer programs over two-layer simplices by triangulation.

Analysis

It is easy to verify that the conditions (C-1) and (C-2) hold for this reduction. Recall that the dimension and the number of constraints are fixed. The vertices of P can be computed in constant time. The same holds for the sorted list of first components. The number of two-layer polytopes P_i is constant, and the vertices of these polytopes, together with their partition V and W can also be computed in constant time. Thus an integer program over a polytope in fixed dimension, which is defined by a fixed number of constraints, can be reduced to a constant number of integer programs over two-layer simplices. The sizes of these simplices are linear in the input size.

Lemma 3. *An IP over a polytope $P \subseteq \mathbb{R}^n$ of size s, defined by a fixed number of constraints in fixed dimension n, can be reduced to a constant number of IP's over two-layer simplices, each of size $O(s)$.*

2.2 Reduction to the parametric lattice width problem

The *parametric lattice width problem* is defined as follows.

Problem (PLW). Given a full-dimensional polytope $P \subseteq \mathbb{R}^n$ and some $K \in \mathbb{N}$, find a parameter π and an integer vector $c \in \mathbb{Z}^n$ such that the width of the truncated polytope $P_\pi = P \cap (x(1) \geqslant \pi)$ and the width of P_π along c satisfy

$$K \leqslant w(P_\pi), w_c(P_\pi) \leqslant 2^{(n+1)/2+2} \cdot \lceil \sqrt{n} \rceil \cdot K, \tag{4}$$

or assert that $w_c(P) \leqslant 2^{(n+1)/2+2} \cdot \lceil \sqrt{n} \rceil \cdot K$.

Let us motivate this concept. Denote the constant $2^{(n+1)/2+2} \cdot \lceil \sqrt{n} \rceil$ by γ. Run an algorithm for PLW on input P and $f_n + 1$. If this returns a parameter π and an integer vector $c \in \mathbb{Z}^n$ such that $f_n + 1 \leqslant w(P_\pi), w_c(P_\pi) \leqslant \gamma(f_n + 1)$, then the optimum solution of the IP must be in the truncated polytope P_π. This follows from the fact that we are searching an integer point with maximal first component, and that the truncated polytope has to contain integer points by the flatness theorem. On the other hand, this truncation P_π is flat along c. Thus the optimum of IP is the largest optimum of the constant number of the $n-1$-dimensional integer programs

$$\max\{x(1) \mid x \in (P \cap (c^T x = \alpha)) \cap \mathbb{Z}^n\}, \tag{5}$$

where $\alpha \in \mathbb{Z} \cap [\min\{c^T x \mid x \in P\}, \max\{c^T x \mid x \in P\}]$. This means that we have reduced the integer programming problem in dimension n to a constant number of integer programming problems in dimension $n-1$. This is also the case if the algorithm for PLW asserts that $w_c(P) \leqslant \gamma K$.

Analysis

If the size of Σ and K is at most s and PLW can be solved in $O(s)$ steps with rational numbers of size $O(s)$, then the integral vector c which is returned has size $O(s)$. Therefore the constant number of $n-1$-dimensional IP's (5) have size $O(s)$. These can then be transformed into a standard IP with $n-1$ variables and a constant number of constraints, in $O(s)$ steps. Consequently we have the following lemma.

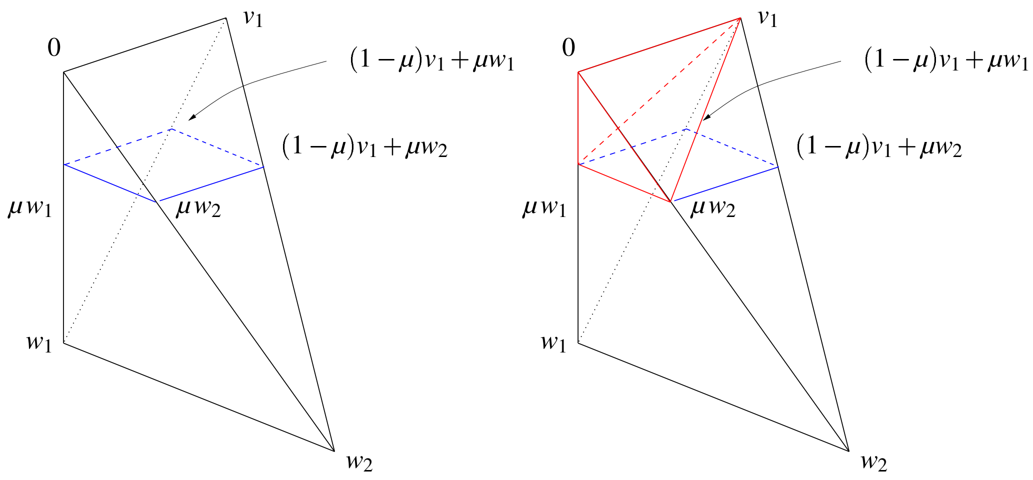

Figure 1: Solving PLW.

Lemma 4. *Suppose that PLW for a two-layer simplex Σ and parameter K with* $\text{size}(\Sigma) + \text{size}(K) = s$ *can be solved with $O(s)$ operations on rational numbers of size $O(s)$, then IP over Σ can also be solved with $O(s)$ operations with rational numbers of size $O(s)$.*

2.3 Reduction to the approximate parametric shortest vector problem

In this section we show how to reduce PLW for a two-layer simplex $\Sigma = \text{conv}(V \cup W)$ and parameter K to an *approximate parametric shortest vector* problem. The width of a polyhedron is invariant under translation. Thus we can assume that $0 \in V$ and that the first component of the vertices in W are negative.

Before we formally describe our approach, let us explain the idea with the help of Figure 1. Here we have a two-layer simplex Σ in 3-space. The set V consists of the points 0 and v_1 and W consists of w_1 and w_2. The picture on the left describes a particular point in time, where the objective function slid into Σ. So we consider the truncation $\Sigma \cap (x(1) \geqslant \pi)$ for some $\pi \geqslant w_1(1)$. This truncation is the convex hull of the points

$$0, v_1, \mu w_1, \mu w_2, (1-\mu)v_1 + \mu w_1, (1-\mu)v_1 + \mu w_2, \tag{6}$$

where $\mu = \pi / w_1(1)$. Now consider the simplex $\Sigma_{V,\mu W}$, which is spanned by the points $0, v_1, \mu w_1, \mu w_2$. This simplex is depicted on the right in Figure 1. If this simplex is scaled by 2, then it contains the truncation $\Sigma \cap (x(1) \geqslant \pi)$. This is easy to see, since the scaled simplex contains the points $2(1-\mu)v_1$, $2\mu w_1$ and $2\mu w_2$. So we have the condition $\Sigma_{V,\mu W} \subseteq \Sigma \cap (x(1) \geqslant \pi) \subseteq 2\Sigma_{V,\mu W}$. From this we can infer the important observation

$$w(\Sigma_{V,\mu W}) \leqslant w(\Sigma \cap (x(1) \geqslant \pi)) \leqslant 2\, w(\Sigma_{V,\mu W}). \tag{7}$$

This means that we can solve PLW for Σ, if we can determine a $\mu \geqslant 0$, such that the width of the simplex $\Sigma_{V,\mu W}$ is sandwiched between K and $(\gamma/2)K$, where γ denotes the constant $2^{(n+1)/2+2} \cdot \lceil \sqrt{n} \rceil$. We now generalize this observation with the following lemma. A proof is straightforward.

Lemma 5. *Let $\Sigma = \operatorname{conv}(V \cup W) \subseteq \mathbb{R}^n$ be a two-layer simplex, where $0 \in V$, $w(1) < 0$ for all $w \in W$ and let π be a number with $0 \geqslant \pi \geqslant w(1)$, $w \in W$. The truncated simplex $\Sigma \cap (x(1) \geqslant \pi)$ is contained in the simplex $2\,\Sigma_{V,\mu W}$, where $\Sigma_{V,\mu W} = \operatorname{conv}(V \cup \mu W)$, where $\mu = \pi / w(1)$, $w \in W$. Furthermore, the following relation holds true*

$$w(\Sigma_{V,\mu W}) \leqslant w\,(\Sigma \cap (x(1) \geqslant \pi)) \leqslant 2\,w(\Sigma_{V,\mu W}). \tag{8}$$

Before we inspect the with of $\Sigma_{V,\mu W}$, let us introduce some notation. We define for an $n \times n$-matrix A, the matrix $A_{\mu,k}$, as

$$A_{\mu,k}(i,j) = \begin{cases} \mu \cdot A(i,j), & \text{if } i \leqslant k, \\ A(i,j), & \text{otherwise}. \end{cases} \tag{9}$$

In other words, the matrix $A_{\mu,k}$ results from A by scaling the first k rows with μ.

Suppose that $V = \{0, v_1, \dots, v_{n-k}\}$ and $W = \{w_1, \dots, w_k\}$. Let $A \in \mathbb{R}^{n \times n}$ be the matrix, whose rows are the vectors $w_1^T, \dots, w_k^T, v_1^T, \dots, v_{n-k}^T$ in this order. The width of $\Sigma_{V,\mu W}$ along the vector c can be bounded as

$$\|A_{\mu,k}\, c\|_\infty \leqslant w_c(\Sigma_{V,\mu W}) \leqslant 2\,\|A_{\mu,k}\, c\|_\infty, \tag{10}$$

and consequently as

$$1/\sqrt{n}\|A_{\mu,k}\, c\| \leqslant w_c(\Sigma_{V,\mu W}) \leqslant 2\,\|A_{\mu,k}\, c\|. \tag{11}$$

This means that we can solve PLW for a two-layer simplex with parameter K if we can determine a parameter $\mu \in \mathbb{Q}_+$ with

$$\lceil \sqrt{n} \rceil \cdot K \leqslant \mathrm{SV}(\Lambda(A_{\mu,k})) \leqslant \gamma/4 \cdot K. \tag{12}$$

Additionally, if $v = A_{\mu,k}\, c$ is a vector of $\Lambda(A_{\mu,k})$ for this parameter μ with $\lceil \sqrt{n} \rceil \cdot K \leqslant \|v\| \leqslant \gamma/4 \cdot K$, then

$$K \leqslant w(\Sigma \cap (x(1) \geqslant \pi)), w_c((x(1) \geqslant \pi)) \leqslant \gamma \cdot K, \tag{13}$$

for $\pi = w_1(1) \cdot \mu$. In other words π and c are a solution to *PLW* for the two-layer simplex Σ.

Thus we can solve PLW for a two-layer simplex $\Sigma = \operatorname{conv}(V \cup W)$ with an algorithm which solves the *approximate parametric shortest vector problem*, which is defined as follows.

Problem (PSV). Given a nonsingular matrix $A \in \mathbb{Q}^{n \times n}$, an integer $1 \leqslant k \leqslant n$, and some $U \in \mathbb{N}$, find a parameter $p \in \mathbb{Q}_{>0}$ such that $U \leqslant \mathrm{SV}(\Lambda(A_{p,k})) \leqslant 2^{(n+1)/2} \cdot U$ or assert that $\mathrm{SV}(\Lambda(A_{p,k})) \leqslant 2^{(n+1)/2} \cdot U$ for all $p \in \mathbb{Q}_{>0}$.

Analysis

We have just described how PLW for a two-layer simplex $\Sigma = \operatorname{conv}(V \cup W)$ and parameter K can be reduced to a PSV. The input to PSV are K, k and the matrix $A \in \mathbb{Q}^{n \times n}$, whose rows are the vectors $w_1^T, \dots, w_k^T, v_1^T, \dots, v_{n-k}^T$. Thus the conditions (C-1) and (C-2) are straightforward.

Lemma 6. *A PLW of size s for a two-layer simplex Σ and parameter K can be reduced to a PSV of size $O(s)$ with $O(s)$ arithmetic operations on rational numbers of size $O(s)$.*

2.4 Solving the parametric shortest vector problem

Suppose we have to solve PSV on input $A \in \mathbb{Q}^{n\times n}, U \in \mathbb{N}$ and $1 \leqslant k \leqslant n$. We argue now that we can assume that A is an integral matrix and that 1 is a lower bound on the parameter p we are looking for. Clearly we can scale the matrix A and U with the product of the denominators of the components of A. In this way we can already assume that A is integral. If A is integral, then $(|\det(A)|, 0, \ldots, 0)$ is an element of $\Lambda(A)$. This implies that we can bound p from below by $1/|\det(A)|$. Thus by scaling U and the last $n-k$ rows of A with $|\det(A)|$, we can assume that $p \geqslant 1$.

The LLL algorithm

In the following, we do not treat the dimension as a constant. First, we briefly review the *LLL-algorithm* for lattice-basis reduction [15]. We refer the reader to the book of Grötschel, Lovász and Schrijver [9] or von zur Gathen and Gerhard [20] for a more detailed account. Intuitively, a lattice basis is reduced, if it is "almost orthogonal". Reduction algorithms apply unimodular transformations of a lattice basis from the right, to obtain a basis whose vectors are more and more orthogonal.

The *Gram-Schmidt orthogonalization* $(b_1^*, \ldots, b_n^*)$ of a basis $(b_1, \ldots, b_n)$ of $\mathbb{R}^n$ satisfies

$$b_j = \sum_{i=1}^{j} \mu_{ji} b_i^*,\ j = 1, \ldots, n, \tag{14}$$

where each $\mu_{jj} = 1$. A lattice basis $B \in \mathbb{Z}^{n\times n}$ is *LLL-reduced*, if the following conditions hold for its Gram-Schmidt orthogonalization.

(i) $|\mu_{i,j}| \leqslant 1/2$, for every $1 \leqslant i < j \leqslant n$;

(ii) $\|b_{j+1}^* + \mu_{j+1,j} b_j^*\|^2 \geqslant 3/4\, \|b_j^*\|^2$, for $j = 1, \ldots, n-1$.

The LLL-algorithm iteratively *normalizes* the basis, which means that the basis is unimodularly transformed into a basis which meets condition (i), and *swaps* two columns if these violate condition (ii). These two steps are repeated until the basis is LLL-reduced. The first column of an LLL-reduced basis is a $2^{(n-1)/2}$-factor approximation to the shortest vector of the lattice.

The key to the termination argument of the LLL-algorithm is the following potential function $\phi(B)$ of a lattice basis $B \in \mathbb{Z}^{n\times n}$:

$$\phi(B) = \|b_1^*\|^{2n} \|b_2^*\|^{2(n-1)} \cdots \|b_1^*\|^2. \tag{15}$$

The potential of an integral lattice basis is always an integer. Furthermore, if B_1 and B_2 are two subsequent bases at the end of the repeat-loop of Algorithm 1, then

$$\phi(B_2) \leqslant \frac{3}{4} \phi(B_1). \tag{16}$$

The potential of the input A can be bounded by $\phi(A) \leqslant (\|a_1\| \cdots \|a_n\|)^{2n}$. The number of iterations can thus be bounded by $O(n(\log \|a_1\| + \ldots + \|a_n\|))$. Step (2) is executed only once and costs $O(n^3)$ operations. The number of operations performed in one iteration of the repeat-loop can be bounded by $O(n^3)$. The rational numbers during the course of the algorithm have polynomial binary encoding length. This implies that the LLL-algorithm has polynomial complexity.

Algorithm 1: LLL
Input: Lattice basis $A \in \mathbb{Z}^{n \times n}$.
Output: Lattice basis $B \in \mathbb{Z}^{n \times n}$ with $\Lambda(A) = \Lambda(B)$ and $\|b_1\| \leqslant 2^{(n-1)/2}\mathrm{SV}(\Lambda(A))$.

$B \leftarrow A$
Compute GSO b_j^*, μ_{ji} of B as in equation (14).
repeat
  foreach $j = 1, \dots, n$
    foreach $i = 1, \dots, j-1$
      $b_j \leftarrow b_j - \lceil \mu_{ji} \rfloor b_i$
  if There is a subscript j which violates condition (ii)
    Swap columns b_j and b_{j+1} of B
  Update GSO b_j^*, μ_{ji}
until B is LLL-reduced
return B

Theorem 7 (Lenstra, Lenstra and Lovász). *Let $A \in \mathbb{Z}^{n \times n}$ be a lattice basis and let A_0 be the number $A_0 = \max\{\|a_j\| \mid j = 1, \dots, n\}$. The LLL-algorithm performs $O(n^4 \log A_0)$ arithmetic operations on rational numbers, whose binary encoding length is $O(n \log A_0)$.*

An algorithm for PSV

We now describe the procedure *iterated LLL*, which solves PSV. The approach is very natural. We use the LLL-algorithm to compute approximate shortest vectors of the lattices $\Lambda(A_{p,k})$ for parameters $p = 2^{\lceil \log U \rceil - i}$ with increasing i, until the approximation of the shortest vector, returned by the LLL-algorithm for $\Lambda(A_{p,k})$ is at most $2^{(n-1)/2}U$.

Algorithm 2: Iterated LLL
Input: Lattice basis $A \in \mathbb{Z}^{n \times n}$, parameters $k, U \in \mathbb{N}$, $1 \leqslant k \leqslant n$.
Output: Paramter p such that $U \leqslant \mathrm{SV}(\Lambda(A_{p,k})) \leqslant 2^{(n+1)/2} \cdot U$.

$p \leftarrow 2^{\lceil \log U \rceil}$
$B \leftarrow A_{p,k}$
repeat
  if $p = 1$
    return $\mathrm{SV}(\Lambda) > U$
  $B \leftarrow B_{1/2,k}$
  $p \leftarrow p/2$
  $B \leftarrow \mathrm{LLL}(B)$
until $\|b_1\| \leqslant 2^{(n-1)/2} \cdot U$
return $2p$

If the sub-lattice Λ' of Λ, which is defined by $\Lambda' = \{v \in \Lambda \mid v(1) = \ldots = v(k) = 0\}$ contains already a nonzero vector v of length $\|v\| < U$, then Algorithm 2 is not correct. We thus make the additional assumption that the LLL-approximation of the shortest vector of Λ' is larger than $2^{(n-1)/2} \cdot U$. If this is not the case, we assert that

$\mathrm{SV}(\Lambda(A_{p,k})) \leqslant 2^{(n+1)/2} \cdot U$ for all $p \in \mathbb{Q}_{>0}$. Notice that a basis of Λ' can be read off the Hermite-Normal-Form of A. Thus the verification of this assumption can be done in the same time, as it is required by one LLL-call.

As the algorithm enters the repeat-loop, we can then be sure that the length of the shortest vector of $\Lambda(A_{p,k})$ is at least U. In the first iteration, this is ensured by the choice of the initial p. In the sequel, this follows, since the shortest vector of $\Lambda(A_{p,k})$ has length at least $\|b_1\|/2^{(n-1)/2} > U$. Consider now the iteration where the condition $\|b_1\| \leqslant 2^{(n-1)/2} \cdot U$ is met. If we scale the first k components of b_1 by 2, we obtain a vector $b' \in \Lambda(A_{2p,k})$. The length of b' satisfies $\|b'\| \leqslant 2 \cdot \|b_1\| \leqslant 2^{(n+1)/2} \cdot U$. On the other hand, we argued above that $\mathrm{SV}(\Lambda_{2p,k}) > U$. Last, if the condition in step (4) is satisfied, then we can assure that $\mathrm{SV}(\Lambda(A)) > U$. This implies the correctness of the algorithm.

Analysis

Let $B^{(0)}, B^{(1)}, \ldots, B^{(s)}$ be the values of B in the course of the algorithm at the beginning of the repeat-loop (step (3)) and consider two consecutive bases $B^{(k)}$ and $B^{(k+1)}$ of this sequence. Step (6) decreases the potential of $B^{(k)}$. Thus by (16), we conclude that the number ℓ of iterations performed by the LLL-algorithm in step (8) satisfies

$$\left(\frac{3}{4}\right)^{\ell} \phi(B^{(k)}) \geqslant \phi(B^{(k+1)}). \tag{17}$$

From this we conclude that the overall amount of iterations through the repeat-loop of the calls to the LLL-algorithm in step (8) can be bounded by

$$O(\log \phi(B^{(0)})) = O(\log \phi(A_{U,k})). \tag{18}$$

The potential $\phi(A_{U,k})$ can be bounded by $\phi(A_{U,k}) \leqslant \phi(A_{U,n}) \leqslant U^{2n^2}(\|a_1\| \cdots \|a_n\|)^{2n}$. As in the analysis of the LLL-algorithm, let A_0 be the number $A_0 = \max\{\|a_j\| \mid i = 1, \ldots, n\}$. The overall number of iterations number of iterations through the repeat-loop of the LLL-algorithm can be bounded by

$$O(n^2(\log U + \log A_0)). \tag{19}$$

Each iteration performs $O(n^3)$ operations. As far as the binary encoding length of the numbers is concerned, we can directly apply Theorem 7 to obtain the next result.

Theorem 8. *Let $A \in \mathbb{Z}^{n \times n}$ be a lattice basis, $U \in \mathbb{N}$ and $1 \leqslant k \leqslant n$ be positive integers. Furthermore let $A_0 = \max\{\|a_j\| \mid j = 1, \ldots, n\}$. The parametric shortest vector problem for A, U and k can be solved with $O(n^5(\log U + \log A_0))$ basic arithmetic operations and comparisons with rational numbers of binary encoding length $O(n(\log A_0 + \log U))$.*

This shows that the complexity of *PSV* in fixed dimension n is linear in the input size and operates on integers whose size is also linear in the input. This concludes the proof of Theorem 1. □

As a consequence, we obtain the following result using Clarkson's [2] random sampling algorithm.

Theorem 9. *An integer program* (1) *in fixed dimension n, where the objective vector and each of the m constraints of $Ax \leqslant b$ have binary encoding length at most s, can be solved with an expected amount of $O(m + \log(m)\,s)$ arithmetic operations on rational numbers of size $O(s)$.*

Acknowledgement

Many thanks are due to Günter Rote for his comments and suggestions on an earlier version of this paper.

References

[1] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 601–610. ACM Press, 2001.

[2] K. L. Clarkson. Las vegas algorithms for linear and integer programming when the dimension is small. *Journal of the Association for Computing Machinery*, 42:488–499, 1995.

[3] P. Domich, R. Kannan, and L. E. Trotter. Hermite normal form computation using modulo determinant arithmetic. *Mathematics of Operations Research*, 12(1):50–59, 1987.

[4] F. Eisenbrand and G. Rote. Fast 2-variable integer programming. In K. Aardal and B. Gerards, editors, *Integer Programming and Combinatorial Optimization, IPCO 2001*, volume 2081 of *LNCS*, pages 78–89. Springer, 2001.

[5] S. D. Feit. A fast algorithm for the two-variable integer programming problem. *Journal of the Association for Computing Machinery*, 31(1):99–113, 1984.

[6] A. Frank and É. Tardos. An application of simultaneous Diophantine approximation in combinatorial optimization. *Combinatorica*, 7:49–65, 1987.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freemann, 1979.

[8] B. Gärtner and E. Welzl. Linear programming—randomization and abstract frameworks. In *STACS 96 (Grenoble, 1996)*, volume 1046 of *Lecture Notes in Comput. Sci.*, pages 669–687. Springer, Berlin, 1996.

[9] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988.

[10] N. Kanamaru, T. Nishizeki, and T. Asano. Efficient enumeration of grid points in a convex polygon and its application to integer programming. *International Journal of Computational Geometry & Applications*, 4(1):69–85, 1994.

[11] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987.

[12] R. Kannan and A. Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM Journal on Computing*, 8(4):499–507, 1979.

[13] R. Kannan and L. Lovász. Covering minima and lattice-point-free convex bodies. *Annals of Mathematics*, 128:577–602, 1988.

[14] D. Knuth. *The art of computer programming*, volume 2. Addison-Wesley, 1969.

[15] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Annalen*, 261:515 – 534, 1982.

[16] H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538 – 548, 1983.

[17] N. Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the Association for Computing Machinery*, 31:114–127, 1984.

[18] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley, 1988.

[19] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley, 1986.

[20] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.

[21] L. Y. Zamanskij and V. D. Cherkasskij. A formula for determining the number of integral points on a straight line and its application. *Ehkon. Mat. Metody*, 20:1132–1138, 1984.

INFORMATIK

Below you find a list of the most recent technical reports of the Max-Planck-Institut für Informatik. They are available by anonymous ftp from `ftp.mpi-sb.mpg.de` under the directory `pub/papers/reports`. Most of the reports are also accessible via WWW using the URL `http://www.mpi-sb.mpg.de`. If you have any questions concerning ftp or WWW access, please contact `reports@mpi-sb.mpg.de`. Paper copies (which are not necessarily free of charge) can be ordered either by regular mail or by e-mail at the address below.

Max-Planck-Institut für Informatik
Library
attn. Anja Becker
Stuhlsatzenhausweg 85
66123 Saarbrücken
GERMANY
e-mail: `library@mpi-sb.mpg.de`

MPI-I-2003-NWG2-002	F. Eisenbrand	Fast integer programming in fixed dimension
MPI-I-2003-NWG2-001	L.S. Chandran, C.R. Subramanian	Girth and Treewidth
MPI-I-2003-4-001	M. Tarini, H.P.R. Lensch, M. Goesele, H. Seidel	3-D Aqusition of Mirrowing Objects
MPI-I-2003-2-002	M. Jaeger	A Representation Theorem and Applications to Measure Selection and Noninformative Priors
MPI-I-2003-2-001	P. Maier	Compositional Circular Assume-Guarantee Rules Cannot Be Sound And Complete
MPI-I-2003-1-010	H. Tamaki	A linear time heuristic for the branch-decomposition of planar graphs
MPI-I-2003-1-009	B. Csaba	On the Bollobás – Eldridge conjecture for bipartite graphs
MPI-I-2003-1-008	P. Sanders	Soon to be published
MPI-I-2003-1-007	H. Tamaki	Alternating cycles contribution: a strategy of tour-merging for the traveling salesman problem
MPI-I-2003-1-006	H. Tamaki, M. Dietzfelbinger	On the probability of Rendezvous in Graph
MPI-I-2003-1-005	M. Dietzfelbinger, P. Woelfel	Almost Random Graphs with Simple Hash Functions
MPI-I-2003-1-004	E. Althaus, T. Polzin, S.V. Daneshmand	Improving Linear Programming Approaches for the Steiner Tree Problem
MPI-I-2003-1-003	R. Beier, B. Vöcking	Random Knapsack in Expected Polynomial Time
MPI-I-2003-1-002	P. Krysta, P. Sanders, B. Vöcking	Scheduling and Traffic Allocation for Tasks with Bounded Splittability
MPI-I-2003-1-001	P. Sanders, R. Dementiev	Asynchronous Parallel Disk Sorting
MPI-I-2002-4-002	F. Drago, W. Martens, K. Myszkowski, H. Seidel	Perceptual Evaluation of Tone Mapping Operators with Regard to Similarity and Preference
MPI-I-2002-4-001	M. Goesele, J. Kautz, J. Lang, H.P.A. Lensch, H. Seidel	Tutorial Notes ACM SM 02 A Framework for the Acquisition, Processing and Interactive Display of High Quality 3D Models
MPI-I-2002-2-008	W. Charatonik, J. Talbot	Atomic Set Constraints with Projection
MPI-I-2002-2-007	W. Charatonik, H. Ganzinger	Symposium on the Effectiveness of Logic in Computer Science in Honour of Moshe Vardi
MPI-I-2002-1-008	P. Sanders, J.L. Träff	The Factor Algorithm for All-to-all Communication on Clusters of SMP Nodes
MPI-I-2002-1-005	M. Hoefer	Performance of heuristic and approximation algorithms for the uncapacitated facility location problem
MPI-I-2002-1-004	S. Hert, T. Polzin, L. Kettner, G. Schäfer	Exp Lab A Tool Set for Computational Experiments
MPI-I-2002-1-003	I. Katriel, P. Sanders, J.L. Träff	A Practical Minimum Scanning Tree Algorithm Using the Cycle Property

MPI-I-2002-1-002	F. Grandoni	Incrementally maintaining the number of l-cliques
MPI-I-2002-1-001	T. Polzin, S. Vahdati	Using (sub)graphs of small width for solving the Steiner problem
MPI-I-2001-4-005	H.P.A. Lensch, M. Goesele, H. Seidel	A Framework for the Acquisition, Processing and Interactive Display of High Quality 3D Models
MPI-I-2001-4-004	S.W. Choi, H. Seidel	Linear One-sided Stability of MAT for Weakly Injective Domain
MPI-I-2001-4-003	K. Daubert, W. Heidrich, J. Kautz, J. Dischler, H. Seidel	Efficient Light Transport Using Precomputed Visibility
MPI-I-2001-4-002	H.P.A. Lensch, J. Kautz, M. Goesele, H. Seidel	A Framework for the Acquisition, Processing, Transmission, and Interactive Display of High Quality 3D Models on the Web
MPI-I-2001-4-001	H.P.A. Lensch, J. Kautz, M. Goesele, W. Heidrich, H. Seidel	Image-Based Reconstruction of Spatially Varying Materials
MPI-I-2001-2-006	H. Nivelle, S. Schulz	Proceeding of the Second International Workshop of the Implementation of Logics
MPI-I-2001-2-005	V. Sofronie-Stokkermans	Resolution-based decision procedures for the universal theory of some classes of distributive lattices with operators
MPI-I-2001-2-004	H. de Nivelle	Translation of Resolution Proofs into Higher Order Natural Deduction using Type Theory
MPI-I-2001-2-003	S. Vorobyov	Experiments with Iterative Improvement Algorithms on Completely Unimodel Hypercubes
MPI-I-2001-2-002	P. Maier	A Set-Theoretic Framework for Assume-Guarantee Reasoning
MPI-I-2001-2-001	U. Waldmann	Superposition and Chaining for Totally Ordered Divisible Abelian Groups
MPI-I-2001-1-007	T. Polzin, S. Vahdati	Extending Reduction Techniques for the Steiner Tree Problem: A Combination of Alternative-and Bound-Based Approaches
MPI-I-2001-1-006	T. Polzin, S. Vahdati	Partitioning Techniques for the Steiner Problem
MPI-I-2001-1-005	T. Polzin, S. Vahdati	On Steiner Trees and Minimum Spanning Trees in Hypergraphs
MPI-I-2001-1-004	S. Hert, M. Hoffmann, L. Kettner, S. Pion, M. Seel	An Adaptable and Extensible Geometry Kernel
MPI-I-2001-1-003	M. Seel	Implementation of Planar Nef Polyhedra
MPI-I-2001-1-002	U. Meyer	Directed Single-Source Shortest-Paths in Linear Average-Case Time
MPI-I-2001-1-001	P. Krysta	Approximating Minimum Size 1,2-Connected Networks
MPI-I-2000-4-003	S.W. Choi, H. Seidel	Hyperbolic Hausdorff Distance for Medial Axis Transform
MPI-I-2000-4-002	L.P. Kobbelt, S. Bischoff, K. Kähler, R. Schneider, M. Botsch, C. Rössl, J. Vorsatz	Geometric Modeling Based on Polygonal Meshes
MPI-I-2000-4-001	J. Kautz, W. Heidrich, K. Daubert	Bump Map Shadows for OpenGL Rendering
MPI-I-2000-2-001	F. Eisenbrand	Short Vectors of Planar Lattices Via Continued Fractions
MPI-I-2000-1-005	M. Seel, K. Mehlhorn	Infimaximal Frames: A Technique for Making Lines Look Like Segments
MPI-I-2000-1-004	K. Mehlhorn, S. Schirra	Generalized and improved constructive separation bound for real algebraic expressions
MPI-I-2000-1-003	P. Fatourou	Low-Contention Depth-First Scheduling of Parallel Computations with Synchronization Variables
MPI-I-2000-1-002	R. Beier, J. Sibeyn	A Powerful Heuristic for Telephone Gossiping
MPI-I-2000-1-001	E. Althaus, O. Kohlbacher, H. Lenhof, P. Müller	A branch and cut algorithm for the optimal solution of the side-chain placement problem
MPI-I-1999-4-001	J. Haber, H. Seidel	A Framework for Evaluating the Quality of Lossy Image Compression
MPI-I-1999-3-005	T.A. Henzinger, J. Raskin, P. Schobbens	Axioms for Real-Time Logics
MPI-I-1999-3-004	J. Raskin, P. Schobbens	Proving a conjecture of Andreka on temporal logic