Chapter 9
The LLL Algorithm and Integer Programming

Karen Aardal and Friedrich Eisenbrand

Abstract The LLL algorithm has proven to be a powerful theoretical and practical tool in many areas of discrete mathematics. In this chapter, we review some structural and algorithmic results involving basis reduction and integer programming.

Introduction

Let $P = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{d}\}$, where the $m \times n$ matrix $\mathbf{A}$ and the m-vector $\mathbf{d}$ are given by integer input. Assume P is bounded and full-dimensional. The *Integer programming feasibility problem* is defined as:

$$\text{Does there exist a vector } \mathbf{x} \in P \cap \mathbb{Z}^n? \tag{9.1}$$

This problem is NP-complete [1, 2] and is related to the *Integer programming optimization problem*,

$$\max\{\mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in P \cap \mathbb{Z}^n\}\,, \tag{9.2}$$

where $\mathbf{c}$ is an n-dimensional vector. We call the problem $\max\{\mathbf{c}^T\mathbf{x} \,|\, \mathbf{x} \in P\}$ the *linear programming relaxation* of (9.2). A *combinatorial optimization problem* is typically an integer optimization problem in which the integer variables take values 0 or 1 only. Well-known examples of combinatorial optimization problems are the subset sum problem, the matching problem and the traveling salesman problem.

In 1981, Lenstra, [3, 4] proved that the integer programming feasibility problem (9.1) can be solved in polynomial time if the dimension n is fixed. The proof was algorithmic, and the main auxiliary algorithm was lattice basis reduction. In the research report [3], a reduction algorithm with polynomial running time for fixed n was used, but in the published version [4], Lenstra used the LLL basis reduction algorithm [5] that had been developed in the meantime.

K. Aardal (✉)
Delft Institute of Applied Mathematics, TU Delft, Mekelweg 4, 2628 CD Delft, The Netherlands and CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands,
e-mail: k.i.aardal@tudelft.nl

Not only was Lenstra's result important in that it answered a prominent open complexity question, but it also introduced geometry of numbers to the field of optimization. Many results, inspired by this paper, have since then been obtained.

The purpose of this paper is to provide a glimpse of some of the important theoretical and computational consequences of the LLL algorithm in relation to integer programming, rather than giving a complete overview of all such results. The interested reader can consult the following references for a thorough treatment of the topic. A good undergraduate level introduction to integer programming is given by Wolsey [6]. Graduate textbooks on integer and combinatorial optimization are Grötschel, Lovász, and Schrijver [7], Nemhauser and Wolsey [8], and Schrijver [9,10]. Cassels [11] is a classical book on the geometry of numbers, while the recent books by Barvinok [12] and by Micciancio and Goldwasser [13] focus on algorithmic aspects. Lattices, representations of lattices, and several problems on lattices wherein basis reduction plays a prominent role are presented in the introductory chapter by Lenstra [14]. Lovász [15] treats basis reduction, integer programming, and classical lattice problems such as the shortest vector problem. Kannan [16] provides a nice overview of topics related to lattices and convex bodies, and Aardal and Eisenbrand [17] review results on integer programming in fixed dimension.

Notation

Vectors and matrices are written in boldface. By $\mathbf{x}_j$, we mean the jth vector in a sequence of vectors. The ith element of a vector $\mathbf{x}$ is denoted by x_i. Element (i, j) of the matrix $\mathbf{A}$ is denoted by A_{ij}. The *Euclidean length* of a vector $\mathbf{x} \in \mathbb{R}^n$ is denoted by $\|\mathbf{x}\|$ and is computed as $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$, where $\mathbf{x}^T$ is the transpose of the vector $\mathbf{x}$.

Let $\mathbf{b}_1, \ldots, \mathbf{b}_l$ be linearly independent vectors in $\mathbb{R}^n$. The set

$$L = \left\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{x} = \sum_{j=1}^{l} \lambda_j \mathbf{b}_j, \lambda_j \in \mathbb{Z},\ 1 \le j \le l\right\} \tag{9.3}$$

is called a *lattice*. The set of vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_l\}$ is called a *lattice basis*. If we want to emphasize that we are referring to a lattice L that is generated by the basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_l)$, then we use the notation $L(\mathbf{B})$.

The *rank* of L, rk L, is equal to the dimension of the Euclidean vector space generated by a basis of L. The *determinant* of L can be computed as $d(L) = \sqrt{\det(\mathbf{B}^T\mathbf{B})}$, where $\mathbf{B}^T$ is the transpose of the matrix $\mathbf{B}$ that is formed by taking the basis vectors as columns. Notice that if $l = n$, i.e., L is full-dimensional, then $d(L) = |\det(\mathbf{B})|$.

Let $L(\mathbf{B})$ be a full-dimensional lattice in $\mathbb{R}^n$ generated by $\mathbf{B}$. Its *dual lattice* $L^*(\mathbf{B})$ is defined as

$$L^*(\mathbf{B}) = \left\{ \mathbf{x} \in \mathbb{R}^n \mid \mathbf{x}^T \mathbf{y} \in \mathbb{R} \text{ for all } \mathbf{y} \in L \right\}.$$

The columns of the matrix $(\mathbf{B}^T)^{-1}$ form a basis for the dual lattice $L^*(\mathbf{B})$. For a lattice L and its dual, we have $d(L) = d(L^*)^{-1}$.

Integer Programming: A Brief Background Sketch

Cutting Planes

The history of integer programming is, compared to many other mathematical subjects, quite brief. The first papers on determining optimal solutions to general integer linear optimization problems were published by Ralph E. Gomory; see, for instance, [18, 19]. It is also interesting to read Gomory's [20] own remarks on how he entered the field and viewed the topic. Gomory, while at Princeton, worked as a consultant for the US Navy, and there he was presented with a problem from the Navy Task Force. It was a linear programming problem with the additional important feature that the answer should be given in integer numbers. After having a thorough look at the problem at hand, Gomory made the following observation: all objective function coefficients are integer, so the optimal value should also be integer. One could solve the linear programming relaxation of the problem first, and if the variable values come out integer, then of course the integer optimum has been found. If not, it is valid to add the restriction that the objective value should be less than or equal to the linear programming objective value rounded down. Gomory describes this as "pushing in" the objective function. After some more thinking, Gomory realized that the same thing can be done with other integer forms as well, and the theory of *cutting planes* was born. Gomory proved the important result that, under certain technical conditions, the integer optimum will be obtained after adding a finite number of the so-called Gomory cutting planes. It is important to notice that an algorithm for solving linear optimization problems had been developed in the 1950s by Dantzig [21], so it was natural to use the linear relaxation as a starting point for solving the integer optimization problem.

In a more problem-specific setting, the idea of cutting planes was introduced by Dantzig, Fulkerson, and Johnson [22, 23], who used this approach to solve a 49-city traveling salesman instance by the combination of linear programming and cutting planes. This approach grew increasingly popular with, for instance, the work on the matching problem by Edmonds [24], the traveling salesman problem by Grötschel [25], and Grötschel and Padberg [26–28], and on the knapsack problem by Balas [29], Hammer et al. [30], and Wolsey [31]. The problem of finding good partial descriptions of the convex hull of feasible solutions to various problems has played a prominent role in the research on integer and combinatorial optimization up to this day.

Branch-and-Bound and Branch-and-Cut

In 1960, Land and Doig [32] introduced branch-and-bound. This is an algorithm for integer optimization that implicitly enumerates solutions. Solving linear programming relaxations is the main engine of the algorithm, and information from these relaxations is used to prune the search, with the aim of avoiding complete enumeration. The algorithm can be illustrated by a search tree as follows. In each node of the tree, the linear relaxation of the problem corresponding to that node is solved. When we start out, we are at the root node, where we solve the linear relaxation of the original problem. Let $\underline{z}$ be the value of the best known integer feasible solution. We can stop investigating further at a certain node k, called *pruning* at node k, if one of the following things happens:

1. The solution of the linear programming relaxation of the subproblem corresponding to node k is integer (prune by optimality). If the solution value is better than $\underline{z}$, then update $\underline{z}$.
2. The linear relaxation at node k is infeasible (prune by infeasibility).
3. The objective function value of the linear relaxation at node k is less than or equal to $\underline{z}$ (prune by bound).

If we cannot prune at node k, we need to *branch*, which simply means that we create two subproblems as follows. Choose a variable that has a fractional value in the optimal linear programming solution. Assume this is variable x_i with current fractional value f. One subproblem is created by adding the constraint $x_i \leq \lfloor f \rfloor$ to the linear relaxation of node k, and the other subproblem is created by adding the constraint $x_i \geq \lceil f \rceil$. In this way, we do not cut off any integer solution. The algorithm continues as long as there are unpruned leaves of the tree. Notice that we can also solve the feasibility problem (9.1) by branch-and-bound by just introducing an arbitrary objective function and terminate the search as soon as a feasible integer solution has been found or when integer infeasibility has been established.

Modern integer programming algorithms use a combination of branch-and-bound and cutting planes, both general and problem-specific, where the cutting planes are used to strengthen the linear relaxation in a selection of the nodes of the search tree. We refer to such algorithms as *branch-and-cut*. Branch-and-cut is not only used in academic research codes, but also in commercial software such as CPLEX [33] and Xpress [34].

Complexity Issues

The early work on integer programming took place before there was a formalization of computational complexity, but it was clear from the very beginning that the number of cutting planes needed in a cutting plane algorithm could grow exponentially, and if we consider branch-and-bound, a 2-dimensional example similar to the one given in Example 1 below, illustrates that a branch-and-bound tree can become arbitrarily deep. With the language of computational complexity at hand, these phenomena could be described more formally. From the cutting plane point-of-view, Karp and Papdimitriou [35] proved that it is not possible to find a concise linear description of the convex hull of feasible solutions for an NP-hard optimization problem unless NP=co-NP. This means that we cannot, a priori, write down such a linear description even if we allow for exponentially sized classes of linear inequalities, such as the subtour elimination constraints for the traveling salesman problem.

Example 1. Consider the integer programming feasibility problem (9.1) with the polytope P, illustrated in Fig. 9.1, as input:

If we solve this feasibility problem by branch-and-bound, we first need to introduce an objective function. Let us choose

$$\max z = x_1 + x_2 \,.$$

If we solve the linear relaxation of our problem, we obtain the vector $(x_1, x_2)^T = (6\frac{4}{5}, 5)$. We illustrate P and some of the constraints (dashed lines) added during branch-and-bound in Fig. 9.1, and the search tree corresponding to the

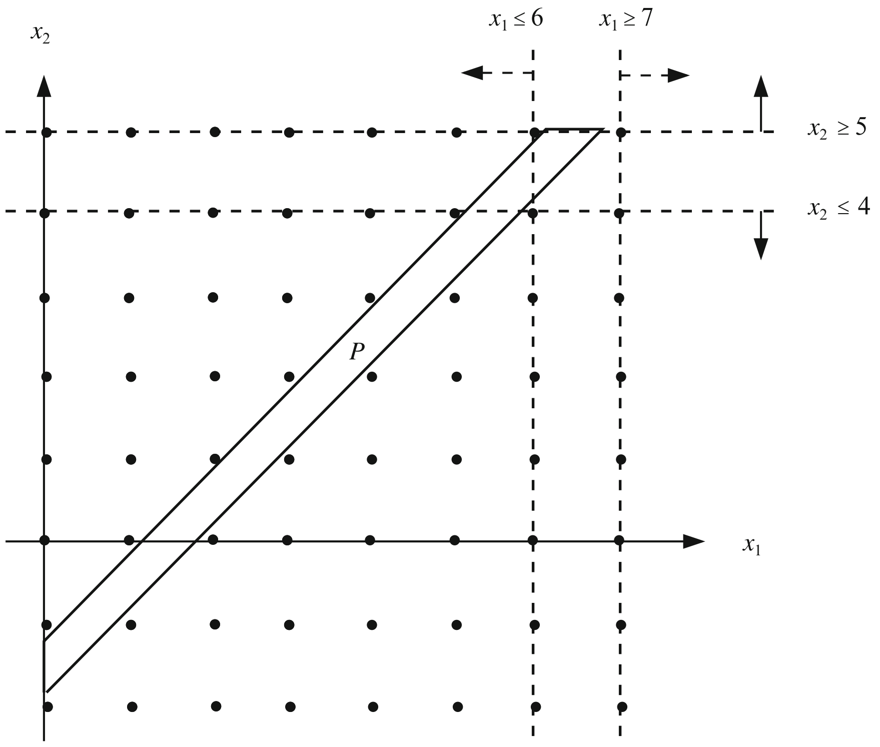

Fig. 9.1 The polytope P of Example 1, and some constraints added in branch-and-bound

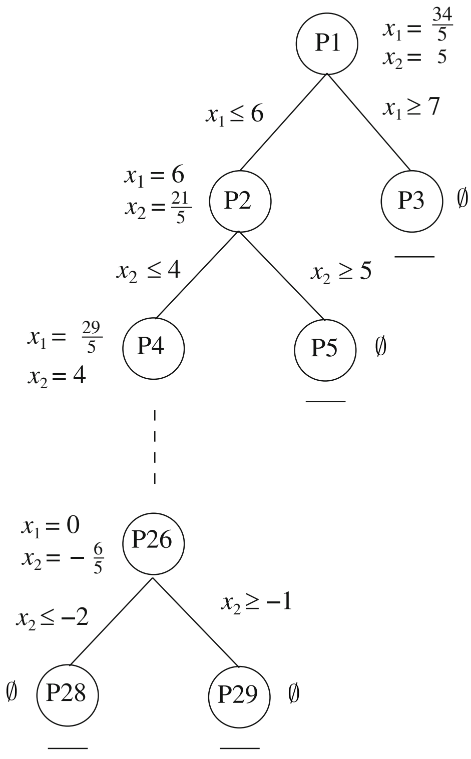

Fig. 9.2 The branch-and-bound search tree

branch-and-bound procedure in Fig. 9.2. Since $(6\frac{4}{5}, 5)$ is not an integer vector, we create two branches at the root node of our search tree: one corresponding to $x_1 \leq 6$ (subproblem P2) and the other corresponding to $x_1 \geq 7$ (subproblem P3). Again, solving the linear relaxation corresponding to subproblem P2 gives the solution $(x_1, x_2)^T = (6, 4\frac{1}{5})$, whereas subproblem P3 is infeasible. Branch-and-bound continues in a similar fashion until subproblems P28 and P29, in which all nodes of the search tree are pruned and it is finally verified that P does not contain any integer vector. □

By "stretching" the polytope given in Example 1 arbitrarily far in both directions, we see that even in dimension $n = 2$, we can obtain a search tree that is arbitrarily deep.

The Integer Linear Feasibility Problem

The above example indicates that branching on variables $x_i \geq \beta$ and $x_i \leq \beta - 1$, for $\beta \in \mathbb{Z}$ can result in an algorithm for integer programming that is exponential

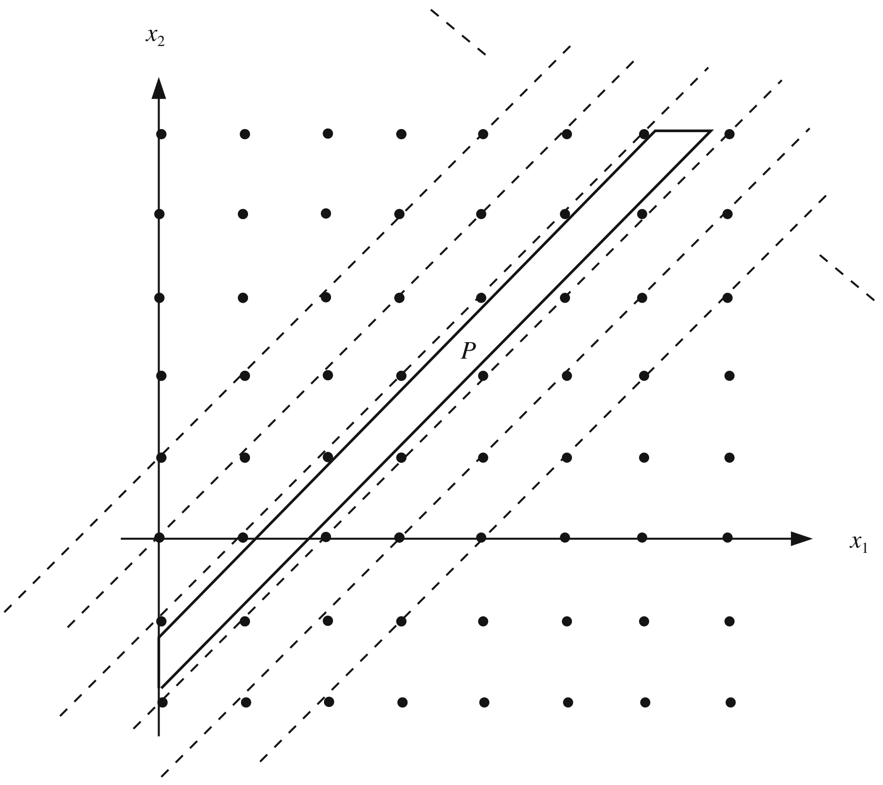

Fig. 9.3 None of the hyperplanes $-x_1 + x_2 = \beta,\ \beta \in \mathbb{Z}$ intersect P

in the binary input encoding of the problem, even in dimension 2. If we allow for hyperplanes that are more general than the single-variable hyperplanes, then we can observe that, for instance, the hyperplanes $-x_1 + x_2 = \beta$, $\beta \in \mathbb{Z}$ do not even intersect with the polytope. Yet, the hyperplanes do contain all points in $\mathbb{Z}^2$. This observation yields a certificate of integer infeasibility of our example; see Fig. 9.3.

The idea of enumerating parallel hyperplanes that cover all lattice points is called *branching on hyperplanes*, and can be described as follows. Let $\mathbf{d} \in \mathbb{Z}^n - \{\mathbf{0}\}$ be a nonzero integer vector. An integer point $\mathbf{x} \in P \cap \mathbb{Z}^n$ satisfies

$$\mathbf{d}^T\mathbf{x} = \beta, \quad \text{where} \quad \beta \in \mathbb{Z} \quad \text{and} \quad \min_{\mathbf{x}\in P} \mathbf{d}^T\mathbf{x} \le \beta \le \max_{\mathbf{x}\in P} \mathbf{d}^T\mathbf{x}.$$

This implies that we can continue to search for an integer point in the lower-dimensional polytopes $P \cap (\mathbf{d}^T\mathbf{x} = \beta)$ for each integer $\beta \in \mathbb{Z}$ satisfying

$$\min_{\mathbf{x}\in P} \mathbf{d}^T\mathbf{x} \le \beta \le \max_{\mathbf{x}\in P} \mathbf{d}^T\mathbf{x}. \tag{9.4}$$

The question is which direction $\mathbf{d}$ to choose such that the number of integers β satisfying (9.4) is small. Clearly, such an integer direction does not need to exist. Simply consider a ball of sufficiently large radius. The *flatness theorem*, attributed to Khinchin 1948, however, ensures that there exists a nonzero integer vector $\mathbf{d} \in \mathbb{Z}^n$ such that the number of integers in the interval 9.4 is bounded by a *constant* if the polytope does not contain an integer point. A *convex body* is a convex and compact set $K \subseteq \mathbb{R}^n$ with a nonempty interior. If we define the *width* of K *along* $\mathbf{d}$ as $w(K, \mathbf{d}) = \max\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in K\} - \min\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in K\}$, the theorem reads as follows.

Theorem 1 (Khinchin's flatness theorem [36]). *Let $K \subseteq \mathbb{R}^n$ be a closed convex set; then, either K contains an integer point, or there exists a nonzero integer vector $\mathbf{d}$ such that $w(K, \mathbf{d}) \leq f(n)$, where $f(n)$ is a constant depending on the dimension only.*

In the following subsection, we will present Lenstra's algorithm as an algorithmic version of the flatness theorem. This is different from the way the algorithm was presented originally, but our presentation below not only links the algorithm explicitly to the flatness theorem, but also highlights the relationship to other traditional lattice problems such as the closest and the shortest vector problems.

Lenstra's Algorithm

Here, whenever we consider a polytope P, we assume it is *full-dimensional*, and we use the notation $\mathbf{d}$ for a *nonzero integer vector* of appropriate dimension.

Lenstra's algorithm finds either an integer point in the polytope $P \subseteq \mathbb{R}^n$, or an integer direction $\mathbf{d}$ such that P is *flat* in this direction, i.e., a direction $\mathbf{d}$ such that $w(P, \mathbf{d})$ is bounded by a constant in fixed dimension. Thus, Lenstra's algorithm solves the following problem, which we call the *Integer feasibility problem (IP)*

Given a polytope $P \subseteq \mathbb{R}^n$, compute an integer point $\mathbf{x} \in P \cap \mathbb{Z}^n$ or a nonzero

$$\text{integer vector } \mathbf{d} \text{ with } w(P, \mathbf{d}) \leq f(n), \tag{9.5}$$

where $f(n)$ is a constant depending on the dimension only.

If problem IP is solvable in polynomial time in fixed dimension, then the integer programming feasibility problem (9.1) is also solvable in polynomial time in fixed dimension. This follows by induction, since in the case in which the algorithm solving IP returns a direction $\mathbf{d}$, one continues the search for an integer point in P in the *constantly* many *lower-dimensional* polytopes

$$P \cap (\mathbf{d}^T\mathbf{x} = \beta), \ \ \beta \in \mathbb{Z}, \ \ \min\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in P\right\} \leq \beta \leq \max\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in P\right\}.$$

In the remainder of this section, we describe Lenstra's result by a series of reductions that ends up with a problem on a lattice L of finding either a lattice vector close to a given vector $\mathbf{u}$, or a short vector in the dual lattice L^*. We call this problem CSVP. In addition, we highlight the role that the LLL algorithm [5] plays in solving the integer programming feasibility problem.

Problem Reductions

An *ellipsoid* is a set $E(\mathbf{C}, \mathbf{c}) = \{\mathbf{x} \in \mathbb{R}^n \mid \|\mathbf{C}(\mathbf{x} - \mathbf{c})\| \leq 1\}$, where $\mathbf{C} \in \mathbb{R}^{n \times n}$ is a nonsingular matrix. In a first step, Lenstra computes an ellipsoid $E(\mathbf{C}, \mathbf{c})$ with $\mathbf{C} \in \mathbb{Q}^{n \times n}$ and $\mathbf{c} \in \mathbb{Q}^n$ such that $E(\mathbf{C}, \mathbf{c})$ is contained in the polytope $P \subseteq \mathbb{Q}^n$ and such that if $E(\mathbf{C}, \mathbf{c})$ is scaled from its center by $2 \cdot n^{3/2}$, then it contains P.

Since the width of the scaled ellipsoid is the width of the original ellipsoid scaled by the same factor, we have

$$w(E(\mathbf{C}, \mathbf{c}), \mathbf{d}) \leq w(P, \mathbf{d}) \leq 2 \cdot n^{3/2} \cdot w(E(\mathbf{C}, \mathbf{c}), \mathbf{d}).$$

This shows that we can solve the problem IP in polynomial time in fixed dimension, if we can solve the following analogous problem for ellipsoids, which we call *Integer feasibility of an ellipsoid (EIP)*, in polynomial time in fixed dimension.

Given a nonsingular rational matrix $\mathbf{C} \in \mathbb{Q}^{n \times n}$ and a rational point $\mathbf{c} \in \mathbb{Q}^n$, compute an integer point $\mathbf{x} \in E(\mathbf{C}, \mathbf{c}) \cap \mathbb{Z}^n$ or determine an integer nonzero vector $\mathbf{d}$ such that $w(E(\mathbf{C}, \mathbf{c}), \mathbf{d}) \leq f_2(n)$,
where $f_2(n)$ is a constant depending on the dimension n only. Following this approach yields $f(n) = 2 \cdot n^{3/2} \cdot f_2(n)$ in (9.5).

In problem EIP, we have to compute a lattice point $\mathbf{v} \in L(\mathbf{C})$, such that its Euclidean distance from the point $\mathbf{Cc}$ is at most 1, or find an integer direction $\mathbf{d}$ such that the ellipsoid is flat along this direction. Since the width along $\mathbf{d}$ of an ellipsoid is invariant under translation of the ellipsoid, one has $w(E(\mathbf{C}, \mathbf{c}), \mathbf{d}) = w(E(\mathbf{C}, \mathbf{0}), \mathbf{d})$.

In other words, if we are not able to find an integer vector $\mathbf{x}$ in $E(\mathbf{C}, \mathbf{c})$ we have to compute an integer direction $\mathbf{d}$ such that

$$\max\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \|\mathbf{Cx}\| \leq 1\right\} - \min\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \|\mathbf{Cx}\| \leq 1\right\} \leq f_2(n)$$

holds. Now, we have

$$\begin{aligned}\max\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \|\mathbf{Cx}\| \leq 1\right\} &= \max\left\{\mathbf{d}^T\mathbf{C}^{-1}\mathbf{Cx} \mid \mathbf{x} \in \mathbb{R}^n,\ \|\mathbf{Cx}\| \leq 1\right\} \\ &= \max\left\{\mathbf{d}^T\mathbf{C}^{-1}\mathbf{y} \mid \mathbf{y} \in \mathbb{R}^n,\ \|\mathbf{y}\| \leq 1\right\} \qquad (9.6) \\ &= \|(\mathbf{C}^T)^{-1}\mathbf{d}\|. \qquad (9.7)\end{aligned}$$

In (9.6) we have used the variable substitution $\mathbf{y} = \mathbf{Cx}$, and in (9.7), we have used the fact that a linear function $\mathbf{f}^T\mathbf{y}$ with $\mathbf{f} \neq \mathbf{0}$ achieves its maximum over the unit ball $B = \{\mathbf{y} \mid \mathbf{y} \in \mathbb{R}^n,\ \|\mathbf{y}\| \leq 1\}$ at the point $\mathbf{y} = \mathbf{f}/\|\mathbf{f}\|$. Similarly, we obtain

$$\min\left\{\mathbf{d}^T\mathbf{x} \mid \mathbf{x} \in \mathbb{R}^n,\ \|\mathbf{Cx}\| \leq 1\right\} = -\|(\mathbf{C}^T)^{-1}\mathbf{d}\|.$$

From this, we can deduce that the width of $E(\mathbf{C},\mathbf{c})$ along an integer direction $\mathbf{d}$ is twice the length of the vector $(\mathbf{C}^T)^{-1}\mathbf{d}$

$$w(E(\mathbf{C},\mathbf{c}),\mathbf{d}) = 2 \cdot \|(\mathbf{C}^T)^{-1}\mathbf{d}\|. \tag{9.8}$$

Next, we observe that the vector $\mathbf{v} = (\mathbf{C}^T)^{-1}\mathbf{d}$ is a lattice vector in the dual lattice

$$L^*(\mathbf{C}) = \left\{(\mathbf{C}^T)^{-1}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^n\right\}$$

of the lattice $L(\mathbf{C})$. Hence, problem EIP has been reduced to the following problem, which we call *problem CSVP*:

Given a nonsingular rational matrix $\mathbf{B} \in \mathbb{Q}^{n\times n}$ and a rational vector $\mathbf{u} \in \mathbb{Q}^n$, compute a lattice vector $\mathbf{x} \in L(\mathbf{B})$ with $\|\mathbf{x} - \mathbf{u}\| \le 1$ or determine a nonzero vector $\mathbf{w} \in L^*(\mathbf{B})$ with $\|\mathbf{w}\| \le f_3(n)$.

In other words, we either have to find a lattice vector *close* to a given vector $\mathbf{u}$, or compute a *short* nonzero vector in the dual lattice. We set $f_2(n) = 2 \cdot f_3(n)$, where the factor of 2 comes from expression (9.8). Tracing back, we have now obtained $f(n) = 2 \cdot n^{3/2} \cdot f_2(n) = 4 \cdot n^{3/2} \cdot f_3(n)$. Notice that finding a short vector in the dual lattice $L^*(\mathbf{B})$ in the Euclidean vector space E is equivalent to finding a hyperplane H in E such that $L(\mathbf{B})$ is contained in widely spaced translates of H; see Lenstra [14].

Using Lenstra's Algorithm to Solve CSVP

Suppose that $\mathbf{B} = (\mathbf{b}_1,\dots,\mathbf{b}_n) \in \mathbb{Q}^{n\times n}$ is a basis of the full-dimensional rational lattice $L(\mathbf{B})$. The *orthogonality defect* of $\mathbf{B}$ is the number $\gamma \in \mathbb{R}$ such that

$$\|\mathbf{b}_1\| \cdot \cdots \cdot \|\mathbf{b}_n\| = \gamma \cdot d(L(\mathbf{B})) = \gamma \cdot |\det(\mathbf{B})|.$$

Notice that $\gamma = 1$ if and only if the basis vectors are pairwise orthogonal. Hermite showed that every lattice in $\mathbb{R}^n$ has a basis $\mathbf{b}_1,\dots,\mathbf{b}_n$ such that $\gamma \le (4/3)^{n(n-1)/4}$, but no polynomial time algorithm is known that can determine a basis with this orthogonality defect guarantee.

Assume further that the longest basis vector is $\mathbf{b}_n$, that is, $\|\mathbf{b}_n\| \ge \|\mathbf{b}_j\|$ for $1 \le j \le n-1$. Let $\mathbf{B}^*$ be the matrix such that

$$\mathbf{B} = \mathbf{B}^* \cdot \mathbf{R},$$

where $\mathbf{R} \in \mathbb{Q}^{n\times n}$ is an upper-triangular matrix with $R_{ii} = 1$ for each $1 \le i \le n$. The matrix $\mathbf{B}^* = (\mathbf{b}_1^*,\dots,\mathbf{b}_n^*)$ is the *Gram-Schmidt orthogonalization* of $\mathbf{B}$. Since we have $\|\mathbf{b}_j\| \ge \|\mathbf{b}_j^*\|$ for each $1 \le j \le n$ and since

$$d(L) = |\det(\mathbf{B})| = |\det(\mathbf{B}^*)| = \|\mathbf{b}_1^*\| \cdot \cdots \cdot \|\mathbf{b}_n^*\|,$$

it follows that $\gamma \geq 1$, which implies the so-called *Hadamard inequality*

$$\|\mathbf{b}_1\| \cdot \cdots \cdot \|\mathbf{b}_n\| \geq |\det(\mathbf{B})|.$$

Our first goal is to find a vector in the lattice $L(\mathbf{B})$ that is close to the given vector $\mathbf{u} \in \mathbb{Q}^n$. Since $\mathbf{B}$ is a basis of $\mathbb{R}^n$, we can write

$$\mathbf{u} = \sum_{j=1}^{n} \lambda_j \mathbf{b}_j,$$

with $\lambda_j \in \mathbb{R}$. The vector

$$\mathbf{v} = \sum_{j=1}^{n} \lfloor \lambda_j \rceil \mathbf{b}_j$$

belongs to the lattice $L(\mathbf{B})$, where $\lfloor \lambda_j \rceil$ denotes the closest integer to λ_j. We have

$$\begin{aligned} \|\mathbf{v} - \mathbf{u}\| &= \| \sum_{i=1}^{n} (\lfloor \lambda_i \rceil - \lambda_i) \mathbf{b}_i \| \leq \sum_{i=1}^{n} \|(\lfloor \lambda_i \rceil - \lambda_i) \mathbf{b}_i\| \leq \frac{1}{2} \sum_{i=1}^{n} \|\mathbf{b}_i\| \\ &\leq \frac{n}{2} \|\mathbf{b}_n\|, \end{aligned} \tag{9.9}$$

where inequality (9.9) holds as the last basis vector $\mathbf{b}_n$ is the longest one in the basis.

If $\|\mathbf{v} - \mathbf{u}\| \leq 1$, we have solved problem CSVP as stated at the end of Section "Problem Reductions". Suppose, therefore, that $\|\mathbf{v} - \mathbf{u}\| > 1$. We now need to find a short vector in the dual lattice $L^*(\mathbf{B})$. From inequality (9.9), we obtain

$$\|\mathbf{b}_n\| \geq 2/n\,. \tag{9.10}$$

If we combine

$$\|\mathbf{b}_1\| \cdot \cdots \cdot \|\mathbf{b}_n\| = \gamma \cdot \|\mathbf{b}_1^*\| \cdot \cdots \cdot \|\mathbf{b}_n^*\|$$

and

$$\|\mathbf{b}_j\| \geq \|\mathbf{b}_j^*\|, \quad 1 \leq j \leq n\,,$$

we obtain $\|\mathbf{b}_n\| \leq \gamma \cdot \|\mathbf{b}_n^*\|$, which together with 9.10 implies

$$\|\mathbf{b}_n^*\| \geq 2/(n \cdot \gamma).$$

The vector $\mathbf{b}_n^*$ is orthogonal to the vectors $\mathbf{b}_j^*$, $1 \leq j \leq n-1$, and since $\mathbf{R}$ is an upper triangular matrix with only 1's on its diagonal, it follows that $(\mathbf{b}_n^*)^T \mathbf{B}^* \cdot \mathbf{R} = (0, \ldots, 0, \|\mathbf{b}_n^*\|^2)$. Next, let $\mathbf{v} = \mathbf{B}\mathbf{x}$, $\mathbf{x} \in \mathbb{Z}$. Notice that $\mathbf{v} \in L(\mathbf{B})$. We now show that the vector $(1/\|\mathbf{b}_n^*\|^2)\mathbf{b}_n^*$ belongs to the dual lattice $L^*(\mathbf{B})$ by showing that $(1/\|\mathbf{b}_n^*\|^2)\mathbf{b}_n^{*T}\mathbf{v} \in \mathbb{Z}$:

$$(1/\|\mathbf{b}_n^*\|^2)\mathbf{b}_n^{*T}\mathbf{v} = (1/\|\mathbf{b}_n^*\|^2)\mathbf{b}_n^{*T}\mathbf{B}^*\mathbf{R}\mathbf{x}$$

$$= (0, \ldots, 0, 1)\,\mathbf{x}$$
$$= x_n \in \mathbb{Z},$$

Hence,

$$\mathbf{w} = (1/\|\mathbf{b}_n^*\|^2)\,\mathbf{b}_n^* \in L^*(\mathbf{B}), \tag{9.11}$$

and the norm of $\mathbf{w}$ satisfies

$$\|\mathbf{w}\| \leq (n \cdot \gamma)/2. \tag{9.12}$$

The length of $\mathbf{w}$ can be bounded by a constant depending only on n if the orthogonality defect γ can be bounded by such a constant.

The Role of the LLL Algorithm for Solving IP

As described above, Lenstra [4] has shown that *any* basis reduction algorithm that runs in polynomial time in fixed dimension and returns a basis, such that its orthogonality defect is bounded by a constant in fixed dimension suffices to solve CSVP in polynomial time in fixed dimension, and consequently the integer feasibility problem for a rational polytope. If the LLL algorithm is applied to reduce the basis $\mathbf{B}$, then $\gamma \leq 2^{n(n-1)/4}$. Our discussion above shows now that IP can be solved in polynomial time with $f(n) = 4 \cdot n^{3/2} \cdot f_3(n) = 2 \cdot n^{5/2} \cdot 2^{n(n-1)/4}$. In fact, this constant can be slightly improved by a better bound for 9.9. More precisely, for a given $\mathbf{u} \in \mathbb{Q}^n$, one can compute, using the Gram-Schmidt orthogonalization of $\mathbf{B}$, a lattice vector $\mathbf{v} \in L(\mathbf{B})$ with

$$\|\mathbf{v} - \mathbf{u}\| \leq (\sqrt{n}/2) \cdot \|\mathbf{b}_n\|.$$

By propagating this improvement through the constants, we obtain the bound $f(n) \leq 2 \cdot n^2 \cdot 2^{n(n-1)/4}$, yielding Lenstra's main result.

Theorem 2 ([3,4]). *Given a rational polytope $P = \{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$, one can compute either an integer point $\mathbf{x} \in P \cap \mathbb{Z}^n$ or a nonzero integer vector $\mathbf{d} \in \mathbb{Z}^n$ with $w(P, \mathbf{d}) \leq 2 \cdot n^2 \cdot 2^{n(n-1)/4}$ in polynomial time. The integer linear feasibility problem can be solved in polynomial time, if the dimension is fixed.*

Related Results

Lovász [15] obtained the following result by combining basis reduction and a different way of obtaining an inscribed ellipsoid. His result is more general in the sense that it applies to convex bodies. Let K be a convex body. The unique maximum-volume ellipsoid that is contained in K is called *Löwner-John ellipsoid*. If this ellipsoid is scaled from its center by a factor of n, then it contains the body K. The Löwner-John ellipsoid can be found with the ellipsoid method [7] in polynomial time, provided one can solve the *weak separation problem* [7] for K in polynomial time.

This, together with the LLL algorithm, yields the following result.

Theorem 3 ([15]). *Let $K \subseteq \mathbb{R}^n$ be a convex body for which one can solve the weak separation problem in polynomial time. We can achieve, in polynomial time, one of the following:*

(i) find an integer vector in X, or
(ii) find an integer vector $\mathbf{c} \in \mathbb{Z}^n$ with

$$\max\left\{\mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in X\right\} - \max\left\{\mathbf{c}^T\mathbf{x} \mid \mathbf{x} \in X\right\} \leq 2 \cdot n^2 \cdot 9^n.$$

The polynomial running time in the above theorem depends on the binary encoding length of the radii of a ball, which is inscribed in K and a ball containing K respectively, see [7].

Lovász and Scarf [37] developed a basis reduction algorithm, called *generalized basis reduction*, based on a polyhedral norm, and used it to solve the integer programming feasibility problem. No polynomial algorithm is known to find a generalized reduced basis in the sense of Lovász and Scarf. Such a basis can, however, be derived in polynomial time if the dimension is fixed. Since a reduced basis can be found by solving a sequence of linear programs, this algorithms is still interesting from the implementation point of view. See also the comments at the end of this section.

The *packing radius* $\rho(L)$ of a lattice $L \subseteq \mathbb{R}^n$ is half the length of the shortest nonzero vector of L. It is the largest number α such that the interior of balls centered at lattice points of radius α does not intersect. The *covering radius* $\mu(L)$ is the smallest number β such that the balls of radius β centered at lattice points cover the whole space $\mathbb{R}^n$. The number $\mu(L)$ is the largest distance of a point in $\mathbb{R}^n$ to the lattice L. The flatness theorem implies that $\rho(L) \cdot \mu(L^*) \leq c(n)$, where $c(n)$ is a constant depending on the dimension n only. Lagarias, Lenstra, and Schnorr [38] have shown that $c(n) \leq n^{3/2}/4$. Banaszczyk [39] proved that $c(n) = O(n)$. This shows that a closest vector in L to a vector $\mathbf{u}$ can be computed with $O\left((c \cdot n)!\right)$ shortest vector queries, where c is some constant. The dependence on the dimension n in Lenstra's algorithm is $O(2^{n^3})$. Kannan [40], see also [16], presented an algorithm for integer programming with running time $n^{O(n)}$. Kannan and Lovász [41] have shown that the constant in Kinchines flatness theorem is $O(n^2)$ if K is a rational polytope. The fastest algorithm to compute a shortest vector is by Ajtai [42] is randomized and has an expected running time of $2^{O(n)}$ times a polynomial in the input encoding of the basis. Blömer [43] presented a deterministic algorithm that computes the closest vector in time $n!$ times a polynomial in the input encoding length of the basis.

Barvinok [44] considered the problem of *counting* integer points in a polytope, which is a generalization of the integer feasibility problem. He used an approach based on an identity of Brion for exponential sums over polytopes. Lenstra's algorithm was used as a subroutine, but later Dyer and Kannan [45] showed, in a modification of Barvinok's algorithm, that this subroutine was in fact not needed.

A topic related to the integer feasibility problem (9.1), is the problem of finding the Hermite normal form of a matrix $\mathbf{A}$. The *Hermite normal form* of a matrix $\mathbf{A} \in \mathbb{Z}^{m\times n}$ of full row rank, HNF($\mathbf{A}$), is obtained by multiplying $\mathbf{A}$ by an $n \times n$ unimodular matrix $\mathbf{U}$ to obtain the form $(\mathbf{D},\ \mathbf{0})$, where $\mathbf{D} \in \mathbb{Z}^{m\times m}$ is a nonsingular, nonnegative lower triangular matrix with the unique row maximum along the diagonal. An integer nonsingular matrix $\mathbf{U}$ is *unimodular* if $\det(\mathbf{U}) = \pm 1$. If the matrix $\mathbf{A}$ is rational, then it has a unique Hermite normal form.

Given is a system of rational equations $\mathbf{Ax} = \mathbf{d}$. The question is whether this system has a solution in integers. Frumkin [46,47] and von zur Gathen and Sieveking [48] showed that solving such a system of linear Diophantine equations can be done in polynomial time. Von zur Gathen and Sieveking, and Votyakov and Frumkin [49] showed that it is possible to find a basis for the lattice $L = \{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{Ax} = \mathbf{0}\}$ is polynomial time. From this result, Frumkin [50] deduced that it is possible to find HNF($\mathbf{A}$) in polynomial time. Kannan and Bachem [51] developed a direct polynomial time algorithm for finding HNF($\mathbf{A}$).

Theorem 4 ([48,49]). *Given a feasible system* $\mathbf{Ax} = \mathbf{d}$ *of rational linear equations, one can find, in polynomial time, integer vectors* $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t$ *such that*

$$\left\{\mathbf{x} \in \mathbb{Z}^n \mid \mathbf{Ax} = \mathbf{d}\right\} = \left\{\mathbf{x}_0 + \sum_{j=1}^{t} \lambda_j \mathbf{x}_j \mid \lambda \in \mathbb{Z}^t\right\} \tag{9.13}$$

Let $\mathbf{AU} = (\mathbf{D},\ \mathbf{0})$ be the Hermite normal form of $\mathbf{A}$. Then, we can choose

$$\mathbf{x}_0 = \mathbf{U}\begin{pmatrix}\mathbf{D}^{-1}\mathbf{d}\\ \mathbf{0}\end{pmatrix}, \quad \mathbf{x}_j = \mathbf{U}\begin{pmatrix}\mathbf{0}\\ \mathbf{e}_j\end{pmatrix}, \ 1 \le j \le t\,.$$

Notice that $\mathbf{Ax}_0 = \mathbf{d}$ and that $\mathbf{Ax}_j = \mathbf{0},\ 1 \le j \le t$.

Schrijver [9], p. 74, discusses how one can use the LLL algorithm to find the Hermite normal form of a matrix; see also [52].

Aardal, Hurkens and Lenstra [53] used the representation (9.13) in which the vectors $\mathbf{x}_j,\ 1 \le j \le t$ are LLL-reduced basis vectors of the lattice $L_0 = \{\mathbf{y} \in \mathbb{Z}^n \mid \mathbf{Ay} = \mathbf{0}\}$, i.e., they use the reformulation

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{B}_0\lambda,\ \lambda \in \mathbb{Z}^t\,, \tag{9.14}$$

where $\mathbf{x}_0$ satisfies $\mathbf{Ax}_0 = \mathbf{d}$, and $\mathbf{B}_0$ is a reduced basis for the lattice L_0. If $\mathbf{A}$ is an $m \times n$ matrix of full row rank, we have $t = n - m$. Aardal et al. obtain the basis $\mathbf{B}_0$ and the vector $\mathbf{x}_0$ by a single application of the LLL algorithm as follows. Consider the system of linear Diophantine equations: $\mathbf{Ax} = \mathbf{d}$ and let $N_1, N_2 \in \mathbb{N}$. Without loss of generality, we assume that $\gcd(a_{i1}, a_{i2}, \dots, a_{in}) = 1$ for $1 \le i \le m$, and that $\mathbf{A}$ has full row rank. Furthermore, let

$$\mathbf{B} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & N_1 \\ N_2\mathbf{A} & -N_2\mathbf{d} \end{pmatrix}, \tag{9.15}$$

and let $\hat{\mathbf{B}}$ be the basis resulting from applying the LLL algorithm to $\mathbf{B}$ in 9.15. The lattice $L(\mathbf{B}) \in \mathbb{R}^{n+m+1}$ is a lattice of rank $n+1$.

Theorem 5 ([53]). *Assume that there exists an integer vector* $\mathbf{x}$ *satisfying the rational system* $\mathbf{Ax} = \mathbf{d}$. *There exist numbers* N_{01} *and* N_{02} *such that if* $N_1 > N_{01}$, *and if* $N_2 > 2^{n+m}N_1^2 + N_{02}$, *then the vectors* $\hat{\mathbf{b}}_j \in \mathbb{Z}^{n+m+1}$ *of the reduced basis* $\hat{\mathbf{B}}$ *have the following properties:*

1. $\hat{b}_{n+1,j} = 0$ *for* $1 \le j \le n-m$,
2. $\hat{b}_{ij} = 0$ *for* $n+2 \le i \le n+m+1$ *and* $1 \le j \le n-m+1$,
3. $|\hat{b}_{n+1,n-m+1}| = N_1$.

Moreover, the sizes of N_{01} *and* N_{02} *are polynomially bounded by the sizes of* $\mathbf{A}$ *and* $\mathbf{d}$.

Theorem 5 implies that if N_1 and N_2 are chosen appropriately, then the first $n-m+1$ columns of the reduced basis $\hat{\mathbf{B}}$ are of the following form:

$$\begin{pmatrix} \mathbf{B}_0 & \mathbf{x}_0 \\ \mathbf{0} & \pm N_1 \\ \mathbf{0} & 0 \end{pmatrix},$$

Aardal and Lenstra [54], and Aardal and Wolsey [55] study the lattice reformulation 9.14 for integer equality knapsack problems in more detail.

To conclude this section, we mention some computational results using LLL-inspired techniques to solve integer programming problems. Gao and Zhang [56] have implemented Lenstra's algorithm. Cook et al. [57] implemented the Lovász-Scarf integer programming algorithm based on generalized basis reduction and reported on computational results of solving several, up to then, unsolved telecommunication network design problems. Aardal, Hurkens, Lenstra [53], Aardal et al. [58], and Aardal and Lenstra [54] report on using the LLL-reduced lattice basis formulation 9.14 and a enumerative algorithm inspired by Lenstra [4] to solve several hard integer feasibility problems.

The Integer Linear Optimization Problem

In this section, we want to consider the integer optimization problem in fixed dimension

$$\max \left\{ \mathbf{c}^T\mathbf{x} \mid \mathbf{Ax} \le \mathbf{b},\, \mathbf{x} \in \mathbb{Z}^n \right\}. \tag{9.16}$$

In the analysis of the algorithms that follow, we use the parameters m and s, where m is the number of inequalities of the system $\mathbf{A}\mathbf{x} \le \mathbf{b}$, and s is an upper bound on the binary encoding length of a coefficient of $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{c}$.

The *greatest common divisor* of two integers a and b can be computed with the Euclidean algorithm with $O(s)$ arithmetic operations, where s is an upper bound on the binary encoding length of the integers a and b. On the other hand, we have the following well-known formula

$$\gcd(a,b) = \min\{a\,x_1 + b\,x_2 \mid a\,x_1 + b\,x_2 \ge 1,\ x_1, x_2 \in \mathbb{Z}\}.$$

This implies that the greatest common divisor can be computed with an algorithm for the integer optimization problem in dimension 2 with one constraint.

The integer optimization problem can be reduced to the integer feasibility problem with binary search. The integer feasibility problem in fixed dimension has complexity $O(m+s)$. This follows from an analysis of Lenstra's algorithm in combination with efficient algorithm to compute a Löwner-John ellipsoid; see [59, 60]. With binary search for an optimal point, one obtains a running time of $O(m \cdot s + s^2)$. If, in addition to the dimension, also the number of constraints is fixed, this results in an $O(s^2)$ algorithm for the integer optimization problem, which is in contrast to the linear running time of the Euclidean algorithm.

Clarkson [61] has shown that the integer optimization problem with m constraints can be solved with an expected number of $O(m)$ arithmetic operations and $O(\log m)$ calls to an oracle solving the integer optimization problem on a constant size subset of the input constraints. Therefore, we concentrate now on the integer optimization problem with a fixed number of constraints. In this section, we outline an algorithm that solves the integer optimization problem in fixed dimension with a fixed number of constraints with $O(s)$ arithmetic operations on rational numbers of size $O(s)$. The algorithm relies on the LLL algorithm.

The first step is to reduce the integer optimization problem over a full-dimensional polytope with a fixed number of facets to a disjunction of integer optimization problems over a constant number of *two-layer simplices*. A two layer simplex is a full-dimensional simplex, whose vertices can be partitioned into two sets V and W, such that the objective function values of the elements in each of the sets V and W agree, i.e., for all $\mathbf{v}_1, \mathbf{v}_2 \in V$, one has $\mathbf{c}^T\mathbf{v}_1 = \mathbf{c}^T\mathbf{v}_2$, and for all $\mathbf{w}_1, \mathbf{w}_2 \in W$, one has $\mathbf{c}^T\mathbf{w}_1 = \mathbf{c}^T\mathbf{w}_2$.

How can one reduce the integer optimization problem over a polytope P to a sequence of integer optimization problems over two-layer simplices? Simply consider the hyperplanes $\mathbf{c}^T\mathbf{x} = \mathbf{c}^T\mathbf{v}$ for each vertex $\mathbf{v}$ of P. If the number of constraints defining P is fixed, then these hyperplanes partition P into a constant number of polytopes, whose vertices can be grouped into two groups, according to the value of their first component. Thus, we can assume that the vertices of P itself can be partitioned into two sets V and W, such that the objective function values of the elements in each of the sets V and W agree. Carathéodory's theorem, see Schrijver [9, p. 94], implies that P is covered by the simplices that are spanned by the vertices of P. These simplices are two-layer simplices. Therefore, the integer optimization problem in fixed dimension with a fixed number of constraints can be reduced in constant time to a constant number of integer optimization problems over a two-layer simplex.

The key idea is then to let the objective function slide into the two-layer simplex, until the width of the truncated simplex exceeds the flatness bound. In this way, one can be sure that the optimum of the integer optimization problem lies in the truncation, which is still flat. Thus, one has reduced the *integer optimization problem* in dimension n to a constant number of *integer optimization problems* in dimension $n-1$, and binary search can be avoided.

How do we determine a parameter π such that the truncated two-layer simplex $\Sigma \cap (\mathbf{c}^T\mathbf{x} \geq \pi)$ just exceeds the flatness bound? We explain the idea with the help of the 3-dimensional example in Fig. 9.4.

Here, we have a two-layer simplex Σ in dimension three. The set V consists of the points $\mathbf{0}$ and $\mathbf{v}_1$ and W consists of $\mathbf{w}_1$, and $\mathbf{w}_2$. The objective is to find a highest point in the vertical direction. The picture on the left describes a particular point in time, where the objective function slid into Σ. So we consider the truncation $\Sigma \cap (\mathbf{c}^T\mathbf{x} \geq \pi)$ for some $\pi \geq \mathbf{c}^T\mathbf{w}_1$. This truncation is the convex hull of the points

$$\mathbf{0}, \mathbf{v}_1, \mu\mathbf{w}_1, \mu\mathbf{w}_2, (1-\mu)\mathbf{v}_1 + \mu\mathbf{w}_1, (1-\mu)\mathbf{v}_1 + \mu\mathbf{w}_2,$$

where $\mu = \pi/\mathbf{c}^T\mathbf{w}_1$. Now consider the simplex $\Sigma_{V,\mu W}$, which is spanned by the points $\mathbf{0}, \mathbf{v}_1, \mu\mathbf{w}_1, \mu\mathbf{w}_2$. This simplex is depicted on the right in Fig. 9.4. If this simplex is scaled by 2, then it contains the truncation $\Sigma \cap (\mathbf{c}^T\mathbf{x} \geq \pi)$. This is easy to see, since the scaled simplex contains the points $2(1-\mu)\,\mathbf{v}_1$, $2\,\mu\,\mathbf{w}_1$ and $2\,\mu\,\mathbf{w}_2$. So we have the condition $\Sigma_{V,\mu W} \subseteq \Sigma \cap (\mathbf{c}^T\mathbf{x} \geq \pi) \subseteq 2\,\Sigma_{V,\mu W}$. From this, we can infer the important observation

$$w(\Sigma_{V,\mu W}) \leq w(\Sigma \cap (\mathbf{c}^T\mathbf{x} \geq \pi)) \leq 2\,w(\Sigma_{V,\mu W}).$$

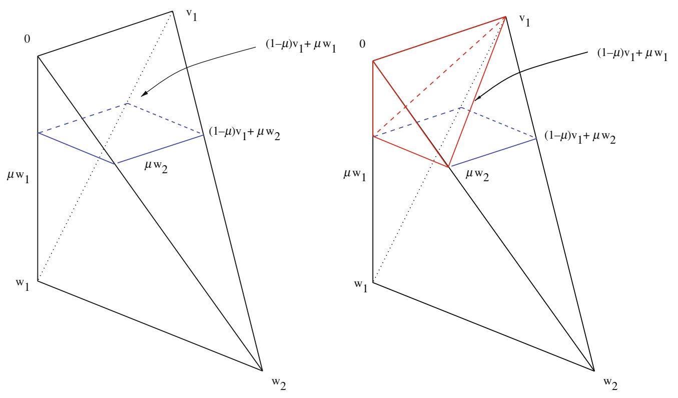

Fig. 9.4 Solving the parametric lattice width problem

This means that we essentially determine the correct π by determining a $\mu \geq 0$, such that the width of the simplex $\Sigma_{V,\mu W}$ just exceeds the flatness bound. The width of $\Sigma_{V,\mu W}$ is roughly (up to a constant factor) the length of the shortest vector of the lattice $L(\mathbf{A}_\mu)$, where $\mathbf{A}_\mu$ is the matrix

$$\mathbf{A}_\mu = \begin{pmatrix} \mu \mathbf{w}_1^T \\ \mu \mathbf{w}_2^T \\ \mathbf{v}_1 \end{pmatrix}.$$

Thus, we have to find a parameter μ, such that the shortest vector of $L(\mathbf{A}_\mu)$ is sandwiched between $f(n)+1$ and $\gamma \cdot (f(n)+1)$ for some constant γ. This problem can be understood as a *parametric shortest vector* problem.

To describe this problem, let us introduce some notation. We define for an $n \times n$-matrix $\mathbf{A} = (a_{ij})_{\forall i,j}$, the matrix $\mathbf{A}^{\mu,k} = (a_{ij})_{\forall i,j}^{\mu,k}$, as

$$a_{ij}^{\mu,k} = \begin{cases} \mu \cdot a_{ij}, & \text{if } i \leq k, \\ a_{ij}, & \text{otherwise.} \end{cases}$$

In other words, the matrix $\mathbf{A}^{\mu,k}$ results from $\mathbf{A}$ by scaling the first k rows with μ. The parametric shortest vector problem is now defined as follows.
Given a nonsingular matrix $\mathbf{A} \in \mathbb{Z}^{n \times n}$ and some $U \in \mathbb{N}$, find a parameter $p \in \mathbb{N}$ such that $U \leq \mathrm{SV}(L(\mathbf{A}^{p,k})) \leq 2^{n+1/2} \cdot U$ or assert that $\mathrm{SV}(L) > U$.
It turns out that the parametric shortest vector problem can be solved in linear time when the dimension is fixed with a cascaded LLL algorithm. From this, it follows that the integer optimization problem in fixed dimension with a fixed number of constraints can be solved in linear time. Together with Clarkson's result, we obtain the following result.

Theorem 6 ([62]). *The integer optimization problem* (9.16) *can be solved with an expected number of* $O(m + s\log m)$ *arithmetic operations on rationals of size* $O(s)$.

Open Problems and Discussion

In the above section, we have sketched a result showing that the integer linear optimization problem can be solved with a linear number of arithmetic operations, if the number of constraints is fixed. The binary encoding length of the numbers in the course of the algorithm remains linear in the input encoding size. Therefore, this result matches the complexity of the Euclidean algorithm if we count arithmetic operations only. When the number m of constraints is arbitrary, Clarkson's algorithm provides a running time of $O(m + s\log m)$, where s is the largest binary encoding length of a coefficient in the input. Clarkson's algorithm is a randomized algorithm. The first question is, whether a deterministic algorithm with running time $O(m + s\log m)$ exists.

In the case of two variables, Eisenbrand and Laue [63] have shown that there exists an algorithm that requires only $O(m + s)$ arithmetic operations. Another question is, whether this result can be extended to any fixed dimension.

The complexity model that reflects the fact that arithmetic operations on large numbers do not come for free is the *bit-complexity* model. Addition and subtraction of s-bit integers take $O(s)$ time. The current state of the art method for multiplication [64] shows that the bit complexity $M(s)$ of multiplication and division is $O(s \log s \log\log s)$.

Recently, Nguyen and Stehlé [65] have presented an LLL-variant that computes an LLL-reduced basis in time $O(n^5(n + \log B)\log B)$ bit-operations, where B is an upper bound on the norm of the vectors in the input. This holds even if the multiplications and divisions are carried out with the straightforward quadratic methods. This means that if the naive algorithms for multiplication and division with remainder are used, the dependence of the running time on the encoding length of the largest binary encoding of a basis-vector component matches exactly the running time of the Euclidean algorithm. In addition, the dependence on the dimension is polynomial. This raises the question, whether these results carry over to the bit-complexity of the integer optimization problem in fixed dimension. In particular, is it possible that this problem can be solved with $O(ms^2)$ bit operations. This would match the complexity of checking whether an integer point is feasible, if the naive methods for multiplication are used.

Acknowledgements This work was partly carried out within the framework of ADONET, a European network in Algorithmic Discrete Optimization, contract no. MRTN-CT-2003-504438. The first author is financed in part by the Dutch BSIK/BRICKS project.

References

1. Borosh, I., Treybig, L.B.: Bounds on positive integral solutions of linear Diophantine equations. Proceedings of the American Mathematical Society **55**, 299–304 (1976)
2. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of Computer Computations, pp 85–103. Plenum Press, NY (1972)
3. Lenstra, Jr., H.W.: Integer programming with a fixed number of variables. Technical Report 81-03, University of Amsterdam, Amsterdam (1981). Available at `http://staff/science/uva.nl/~peter/mi8103/mi8103c.html`
4. Lenstra, Jr., H.W.: Integer programming with a fixed number of variables. Mathematics of Operations Research **8(4)**, 538–548 (1983)
5. Lenstra, A.K., Lenstra, Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mahematische Annalen **261**, 515–534 (1982)
6. Wolsey, L.A.: Integer Programming. Wiley, New York (1998)
7. Grötschel, M. Lovász, L., Schrijver, A.: Geometric Algorithms and Combinatorial Optimization. Springer, Berlin (1988)
8. Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. Wiley, New York (1988)
9. Schrijver, A.: Theory of Linear and Integer Programming. Wiley, Chichester (1986)
10. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency (3 volumes). Algorithms and Combinatorics **24**. Springer, Berlin (2003)

11. Cassels, J.W.S.: An Introduction to the Geometry of Numbers. Classics in Mathematics. Springer, Berlin (1997). Second Printing, Corrected, Reprint of the 1971 ed.
12. Barvinok, A.: A Course in Convexity. Graduate Studies in Mathematics **54**. American Mathematical Society, Providence, RI (2002)
13. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems: A Cryptographic Perspective. The Kluwer International Series in Engineering and Computer Science **671**. Kluwer Academic Publishers, Boston, Massachusetts (2002)
14. Lenstra, Jr., H.W.: Lattices. Chapter 6 in Algorithmic Number Theory, Mathematical Sciences Research Institute Publications, Vol 44, Cambridge University Press, Cambridge, UK, 127–181, 2008.
15. L. Lovász. An Algorithmic Theory of Numbers, Graphs and Convexity. SIAM, Philadelphia, PA (1986)
16. Kannan, R.: Algorithmic geometry of numbers. Annual Review of Computer Science **2**, 231–267 (1987)
17. Aardal, K., Eisenbrand, F.: Integer programming, lattices and results in fixed dimension. In: Aardal, K., Nemhauser, G.L., Weismantel, R. (eds) Handbook on Discrete Optimization, Chapter 4. North Holland, Amsterdam (2005)
18. Gomory, R.E.: Outline of an algorithm for integer solutions to linear programs. Bulletin of the American Mathematical Society **64**, 275–278 (1958)
19. Gomory, R.E.: An algorithm for integer solutions to linear programs. In: Graves, R.L., Wolfe, P. (eds) Recent Advances in Mathematical Programming, pp 269–302. McGraw-Hill (1963)
20. Gomory, R.E.: Early integer programming. In: Lenstra, J.K., Rinnooy Kan, A.H.G., Schrijver, A. (eds) History of Mathematical Programming: A Collection of Personal Reminiscences, pp 55–61. CWI and North-Holland, Amsterdam (1991)
21. Dantzig, G.B.: Maximization of a linear function of variables subject to linear inequalities. In: Koopmans, T.C. (ed) Activity Analysis of Production and Allocation, pp 339–347. John Wiley & Sons, New York (1951)
22. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: Solution of a large-scale traveling-salesman problem. Operations Research **2**, 393–410 (1954)
23. Dantzig, G.B., Fulkerson, D.R., Johnson, S.M.: On a linear-programming, combinatorial approach to the traveling-salesman problem. Operations Research **7**, 58-66 (1959)
24. Edmonds, J.: Paths, trees and flowers. Canadian Journal of Mathematics **17**, 449–467 (1965)
25. Grötschel, M.: On the symmetric traveling salesman problem: Solution of a 120-city problem. Mathematical Programming Study **12**, 61–77 (1980)
26. Grötschel, M., Padberg, M.W.: Partial linear characterizations of the asymmetric traveling salesman problem. Mathematical Programming **8**, 378–381 (1975)
27. Grötschel, M., Padberg, M.W.: On the symmetric traveling salesman problem I: Inequalities. Mathematical Programming **16**, 265–280 (1978)
28. Grötschel, M., Padberg, M.W.: On the symmetric traveling salesman problem I: Lifting theorems and facets. Mathematical Programming **16**, 281–302 (1978)
29. Balas, E.: Facets of the knapsack polytope. Mathematical Programming **8**, 146–164 (1975)
30. Hammer, P.L., Johnson, E., Peled, U.N.: Facets of regular 0-1 polytopes. Mathematical Programming **8**, 179–206 (1975)
31. Wolsey, L.A.: Faces for a linear inequality in 0-1 variables. Mathematical Programming **8**, 165–178 (1975)
32. Land, A., Doig, A.: An automatic method of solving discrete programming problems. Econometrica **28**, 497–520 (1960)
33. ILOG. Cplex. `http://www.ilog.com/products/cplex`
34. Dash Optimization. Xpress-mp optimization software.
`http://www.dashoptimization.com/home/index.html`
35. Karp, R.M., Papadimitriou, C.H.: On linear characterizations of combinatorial optimization problems. In: 21st Annual Symposium on Foundations of Computer Science, Syracuse, N.Y., pp 1–9. IEEE, New York (1980)
36. Khinchine, A.: A quantitative formulation of Kronecker's theory of approximation (in russian). Izvestiya Akademii Nauk SSR Seriya Matematika **12**, 113–122 (1948)

37. Lovász, L., Scarf, H.E.: The generalized basis reduction algorithm. Mathematics of Operations Research **17(3)**, 751–764 (1992)
38. Lagarias, J., Lenstra, Jr., H.W., Schnorr, C.: Korkin-zolotarev bases and successive minima of a lattice and its reciprocal lattice. Combinatorica **10(4)**, 333–348 (1990)
39. Banaszczyk, W.: Inequalities for convex bodies and polar reciprocal lattices in $\mathbb{R}^n$. II. Application of K-convexity. Discrete Computational Geometry **16(3)**, 305–311 (1996)
40. Kannan, R.: Minkowski's convex body theorem and integer programming. Mathematics of Operations Research **12(3)**, 415–440 (1987)
41. Kannan, R., Lovász, L.: Covering minima and lattice-point-free convex bodies. Annals of Mathematics **128**, 577–602 (1988)
42. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. Proceedings of the 33rd Annual ACM symposium on Theory of Computing, pp 601–610. ACM Press, New York (2001)
43. Blömer, J.: Closest vectors, successive minima, and dual HKZ-bases of lattices. In: Montanari, U., Rolim, J.D.P., Welzl, E. (eds) Automata, Languages and Programming, 27th International Colloquium, ICALP 2000, Geneva, Switzerland, July 9–15, 2000, Proceedings. Lecture Notes in Computer Science **1853**, pp 248–259. Springer, Berlin (2000)
44. Barvinok, A.I.: A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. Mathematics of Operations Research **19(4)**, 769–779 (1994)
45. Dyer, M.E., Kannan, R.: On Barvinok's algorithm for counting lattice points in fixed dimension. Mathematics of Operations Research **22(3)**, 545–549 (1997)
46. Frumkin, M.A.: Algorithms for the solution in integers of systems of linear equations. In: Fridman, A.A. (ed) Studies in discrete optimization (Russian), pp 97–127, Izdat. "Nauka", Moscow (1976)
47. Frumkin, M.A.: An application of modular arithmetic to the construction of algorithms for the solution of systems of linear equations. Doklady Akademii Nauk SSSR **229(5)**, 1067–1070 (1976) [English translation: Soviet Mathematics Doklady **17**, 1165–1168 (1976)]
48. Gathen, von zur, J., Sieveking, M.: Weitere zum Erfüllungsproblem polynomial äquivalente kombinatorische Aufgaben. In: Specker, E. Strassen, V. (eds) Komplexität von Entscheidungsproblemen: Ein Seminar, Lecture Notes in Computer Science **43**, pp 49–71. Springer, Berlin (1976)
49. Votjakov, A.A., Frumkin, M.A.: An algorithm for finding the general integer solution of a system of linear equations. In: Studies in discrete optimization (Russian), pp 128–140. Izdat. "Nauka", Moscow (1976)
50. Frumkin, M.A.: An algorithm for the reduction of a matrix of integers to triangular form with power complexity of the computations. Èkonomika i Matematicheskie Metody **12(1)**, 173–178 (1976)
51. Kannan, R., Bachem, A.: Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. SIAM Journal on Computing **8(4)**, 499–507 (1979)
52. Havas, G., Majewski, B.S., Matthews, K.R.: Extended GCD and Hermite normal form algorithms via lattice basis reduction. Experimental Mathematics **7(2)**, 125–136 (1998) (Addenda and errata: Experimental Mathematics **8**, 179–206)
53. Aardal, K., Hurkens, C.A.J., Lenstra, A.K.: Solving a system of linear Diophantine equations with lower and upper bounds on the variables. Mathematics of Operations Research **25(3)**, 427–442 (2000)
54. Aardal, K.A., Lenstra, A.K.: Hard equality constrained integer knapsacks. Mathematics of Operations Research, **29(3)**, 724–738 (2004). Erratum: Mathematics of Operations Research **31(4)**, 846 (2006)
55. Aardal, K., Wolsey, L.A.: Lattice based extended formulations for integer linear equality systems. Mathematical Programming **121**, 337–352 (2010).
56. Gao, L., Zhang, Y.: Computational experience with Lenstra's algorithm. Technical Report TR02-12, Department of Computational and Applied Mathematics, Rice University, Houston, TX (2002)

57. Cook, W., Rutherford, T., Scarf, H.E., Shallcross, D.: An implementation of the generalized basis reduction algorithm for integer programming. ORSA Journal on Computing **5(2)**, 206–212 (1993)
58. Aardal, K., Bixby, R.E., Hurkens, C.A.J., Lenstra, A.K., Smeltink, J.W.: Market split and basis reduction: Towards a solution of the Cornuéjols-Dawande instances. INFORMS Journal on Computing **12(3)**, 192–202 (2000)
59. Matoušek, J., Sharir, M., Welzl, E.: A subexponential bound for linear programming. Algorithmica **16(4–5)**, 498–516 (1996)
60. Welzl, E.: Smallest enclosing disks (balls and ellipsoids). In: New results and new trends in computer science (Graz, 1991), Lecture Notes in Computer Science **555**, pp 359–370. Springer, Berlin (1991)
61. Clarkson, K.L.: Las Vegas algorithms for linear and integer programming when the dimension is small. Journal of the Association for Computing Machinery **42**, 488–499 (1995)
62. Eisenbrand, F.: Fast integer programming in fixed dimension. In: Battista, G.D., Zwick, U. (eds) Algorithms – ESA 2003. Lecture Notes in Computer Science **2832**, 196–207. Springer, Berlin (2003)
63. Eisenbrand, F., Laue, S.: A linear algorithm for integer programming in the plane. Mathematical Programming **102(2)**, 249 – 259 (2005)
64. Schönhage, A., Strassen, V.: Schnelle Multiplikation grosser Zahlen (Fast multiplication of large numbers). Computing **7**, 281–292 (1971)
65. Nguyen, P.Q., Stehlé, D.: Floating-point LLL revisited. In: Cramer, R. (ed) Advances in Cryptology — EUROCRYPT 2005. Lecture Notes in Computer Science **3494**, pp 215–233. Springer, Berlin (2003)