BASIS REDUCTION METHODS

Gábor Pataki
Department of Statistics and Operations Research, UNC Chapel Hill, Chapel Hill North Carolina

Mustafa Tural
Institute for Mathematics and Its Applications, University of Minnesota, Minneapolis, Minnesota

INTRODUCTION

Consider the integer programming (IP) feasibility problem in the form

$$\text{Find } x \in \mathbb{Z}^n : x \in P, \qquad \text{(IP)}$$

where P is a polyhedron described by inequalities. The work of Lenstra in 1979 [1] answered one of the most challenging questions in the theory of integer programming by presenting an algorithm to solve (IP), whose running time is polynomial, when n, the dimension is fixed. This paper pioneered the use of lattice basis reduction in integer programming, and initiated an interest in polynomial results in integer programming under the "fixed dimension" assumption. Somewhat later Kannan developed an improved variant [2,3], which—to date—has the best theoretical complexity for integer programming feasibility.

The goal of this survey is to review lattice-based methods to solve (IP), focusing on Lenstra's and Kannan's algorithms, which are by now considered "classical," and the more recent reformulation methods of Aardal *et al*. [4], and Krishnamoorthy and Pataki [5].

Example 1. As motivation, let us consider a "hard" IP feasibility problem

$$\begin{array}{rcl} 460 \leq & 51x_1 + 49x_2 & \leq 489 \\ 0 \leq & x_1, x_2 & \leq 10 \\ & x_1, x_2 & \in \mathbb{Z}, \end{array} \qquad (1)$$

shown in Fig. 1. One can see by inspection that it is infeasible.

Let us denote by P the underlying polyhedron. The hyperplanes

$$\left\{ x \,|\, x_1 = k \right\} (k = 0, \ldots, 9)$$

all intersect P, so branch-and-bound trying to prove infeasibility generates 10 subproblems, when branching on x_1. Branching on x_2 also leads to 10 subproblems.

We will return to this example later, in particular, in the section titled "Reformulation Methods" we show that the rangespace reformulation of Krishnamoorthy and Pataki [5] creates an equivalent feasibility problem, in which the set $\left\{ y \,|\, y_2 \in \mathbb{Z} \right\}$ has empty intersection with the underlying polyhedron, hence branching on y_2 solves the problem at the root node.

A lattice L is the set of integral combinations of linearly independent vectors defined as

$$L = \mathbb{L}(B) = \left\{ Bx \,:\, x \in \mathbb{Z}^r \right\} \subseteq \mathbb{R}^n. \qquad (2)$$

The matrix B has r independent columns, which are said to form a *basis* of $\mathbb{L}(B)$. We call r the *dimension* of L, and when $r = n$ we say that L is *full dimensional*. Writing $b_1, \ldots, b_r \in \mathbb{R}^n$ for the columns of B, we also denote L by $\mathbb{L}(b_1, \ldots, b_r)$.

The common theme in lattice-based methods is transforming (IP) into a problem of the form

$$\text{Find } y \in \mathbb{Z}^r : By \in Q, \qquad (\text{IP}_{B,Q})$$

where the matrix B and the polyhedron Q are suitably chosen, so the problem of finding a point in $P \cap \mathbb{Z}^n$ is translated into finding one

Wiley Encyclopedia of Operations Research and Management Science, edited by James J. Cochran
Copyright © 2010 John Wiley & Sons, Inc.

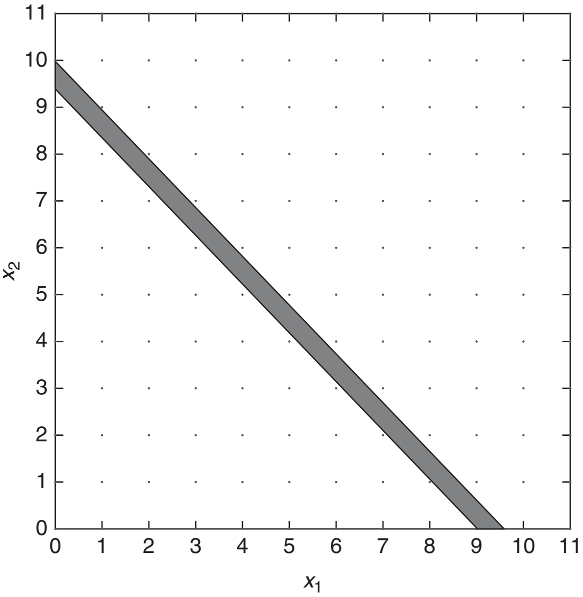

Figure 1. The instance of Example 1.

in $Q \cap \mathbb{L}(B)$. We emphasize that the choice of B and Q is specific to each method, but in all of them, the polyhedron Q has favorable geometry, and B has short and near orthogonal columns, which form a *reduced* basis of $\mathbb{L}(B)$ in a sense to be made precise later. These two facts will allow us to prove good complexity bounds on solving ($\text{IP}_{B,Q}$).

The classical (Lenstra's and Kannan's) algorithms are recursive. They branch on the last few variables in ($\text{IP}_{B,Q}$), that is, enumerate all possible integer values that these can attain, then construct new B matrices and Q polyhedra for each of the resulting subproblems, and so on. However, Mehrotra and Li [6] recently presented a restructuring of the computations in Lenstra's algorithm, so that branching on the last variable in ($\text{IP}_{B,Q}$) is translated back to branching on a suitable hyperplane in the original problem. In contrast, the reformulation methods run a full branch-and-bound algorithm on ($\text{IP}_{B,Q}$), and are implemented by simply feeding the formulation to a commercial IP solver.

The *nullspace reformulation method*, which is applicable to equality constrained integer programs was proposed by Aardal *et al.* [4]. Later, Krishnamoorthy and Pataki [5] introduced the *rangespace reformulation method* for general integer programs. One can analyze these methods on a family of knapsack feasibility problems, with the coefficient vector of the knapsack constraint decomposing as $a = \lambda p + r$, where p and r are integral vectors, and λ is a large integer. There are a variety of instances that fit into this framework with three interesting properties: (i) they are difficult for branch-and-bound that branches on the x_i variables; (ii) their infeasibility is proven by branching on px; and (iii) when λ is sufficiently large, branching on the last variable in the reformulations has the same effect as branching on px in the original problem.

There is a compelling history of these instances, dating back to Jeroslow's famous problem from 1974 [7]. An analysis of nullspace reformulation assuming such decomposable structure was given by Aardal and Lenstra [8,9]. Krishnamoorthy and Pataki [5] showed how a variety of knapsack problems starting with Jeroslow's problem satisfy properties (i) and (ii), presented a "recipe" to generate such instances, and proved Property (iii) for both reformulations.

A recent result of Pataki *et al.* [10] shows that when branch-and-bound is applied to the reformulation of bounded integer programs, the majority of instances get solved without generating any subproblems, that is, at the root node. This result may seem counterintuitive; however, it fits in nicely with the previous work on "low density" subset sum problems, which have been widely used in cryptography.

The rest of the survey is divided into five sections. In the rest of the introduction we describe key definitions that will be used throughout the paper. In the section titled "Reduced Bases" we describe reduced bases of lattices. In the section titled "The Geometry of Lattices and Convex Sets" we present a lemma due to Lenstra on the geometry of lattices, and convex sets, which will play a role in the analysis of both the classical, and the reformulation methods. In the section titled "Lenstra's and Kannan's Algorithms" we review Babai's improved version [11] of Lenstra's algorithm, and Kannan's algorithm. Kannan's algorithm is less frequently covered in surveys than Lenstra's method, perhaps because it is more technical: we think that we succeeded in giving a simplified treatment. In the section titled "Reformulation Methods" we review the reformulation methods, with their complexity analyses. In the section titled "Further Reading and Computational Testing" we point to further reading, and review computational results obtained by lattice-based methods.

The emphasis of the survey is providing simple proofs of complexity results, illustrative examples, and a unifying view of the classical and the reformulation methods. For instance, we will continue Example 1, and show how Lenstra's algorithm, and the rangespace reformulation handle this "difficult" instance. We include exercises, and we think that the survey will be suitable to teach a two- to three-class long segment on lattice-based methods in a course on Integer Programming.

A reader may be interested either only in Lenstra's and Kannan's algorithms, or only in the reformulation methods; so for convenience, sections titled "Lenstra's and Kannan's Algorithms" and "Reformulation Methods" can be read independently of each other.

Exercise 1. Generalize Example 1 by showing that for positive integers λ and μ with $\lambda \geq 2\mu + 1$, the integer programming problem

$$\begin{array}{rcl} \lambda\mu - \lambda + \mu \leq & (\lambda+1)x_1 + (\lambda-1)x_2 & \leq \lambda\mu - \mu - 1 \\ 0 \leq & x_1, x_2 & \leq \mu \\ & x_1, x_2 \in \mathbb{Z} & \end{array} \tag{3}$$

is infeasible, and branching either on x_1 or x_2 will generate at least μ subproblems.

Important Definitions

Branch-and-bound, which we abbreviate as B&B, is a classical solution method for (IP); it was first proposed by Land and Doig [12]. It starts with P as the sole subproblem (node). In a general step, one chooses a subproblem P', a variable x_i, and creates nodes $P' \cap \{x | x_i = \gamma\}$, where γ ranges over all possible integer values of x_i. We repeat this until all subproblems are shown to be empty, or we find an integral point in one of them.

Sometimes we will *branch on hyperplanes* to solve (IP): given a nonzero integral vector p, we let

$$\begin{aligned} k_{\max} &= \max \left\{ px \,|\, x \in P \right\}, \\ k_{\min} &= \min \left\{ px \,|\, x \in P \right\}, \end{aligned} \tag{4}$$

and create the subproblems

$$P \cap \{x \,|\, px = \lfloor k_{\max} \rfloor\}, \ldots, P \cap \{x \,|\, px = \lceil k_{\min} \rceil\}.$$

The number of subproblems is the *integer width* of P along p, that is,

$$\text{iwidth}(p, P) = \lfloor k_{\max} \rfloor - \lceil k_{\min} \rceil + 1.$$

In particular, if $\text{iwidth}(p, P) = 0$, then (IP) is infeasible, and we say that *its infeasibility is proven by branching on px*. Analogously, the *width* of P along p is defined as

$$\text{width}(p, P) = k_{\max} - k_{\min}.$$

To emphasize the difference between Land and Doig's branch-and-bound algorithm, and branching on hyperplanes, we call the former method *ordinary branch-and-bound*.

It is well known [13] that if A is a rational matrix with n columns, then

$$\mathbb{N}(A) := \left\{ x \in \mathbb{Z}^n \,|\, Ax = 0 \right\} \tag{5}$$

is also a lattice, and we call this set the *null-lattice* of A.

We call a matrix U *unimodular*, if it is square, integral, and has determinant ± 1.

For a convex set Q we write $r(Q)$ for the radius of the largest ball in the affine hull of Q that can be inscribed in Q, and $R(Q)$ for the radius of the smallest ball that contains Q:

$$r(Q) = \max\left\{ s \mid \exists p \in Q \text{ s.t. } \mathrm{Ball}(p,s) \cap \mathrm{aff}(Q) \subseteq Q \right\}, \text{ and} \tag{6}$$

$$R(Q) = \min\left\{ s \mid \exists p \in Q \text{ s.t. } \mathrm{Ball}(p,s) \supseteq Q \right\}, \tag{7}$$

where $\mathrm{Ball}(p,s)$ is the Euclidean ball with center p and radius s.

REDUCED BASES

A lattice $\mathbb{L}(B)$ with dimension $r \geq 2$ has infinitely many bases, and it is well known, that all of them are of the form BU, where U is a unimodular matrix. Multiplying B by a unimodular matrix is equivalent to performing a sequence of three elementary column operations on B : multiplying a column by -1; exchanging two columns; and adding an integer multiple of a column to another.

As a lattice may not have an orthonormal basis of unit vectors, we will be interested in bases comprising “reasonably” short and “near orthogonal” ones. These bases will be called *reduced*.

Example 2. Consider the lattice generated by the vectors

$$b_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad b_2 = \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \end{pmatrix}. \tag{8}$$

Multiplying the matrix $[b_1, b_2]$ with

$$U = -\begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$$

gives another basis

$$b_1' = -2b_1 - b_2, \;\; b_2' = -b_1 - b_2.$$

Figure 2 shows the lattice points, and the two bases, with the vectors in the first clearly shorter, and closer to being orthogonal.

We describe lattice bases that are reduced in the sense of Lenstra, Lenstra, and Lovász (LLL), and Korkin and Zolotarev (KZ). We will say that $b_1, \ldots, b_r$ is an LLL (KZ) reduced basis, if it is such a basis of $\mathbb{L}(b_1, \ldots, b_r)$.

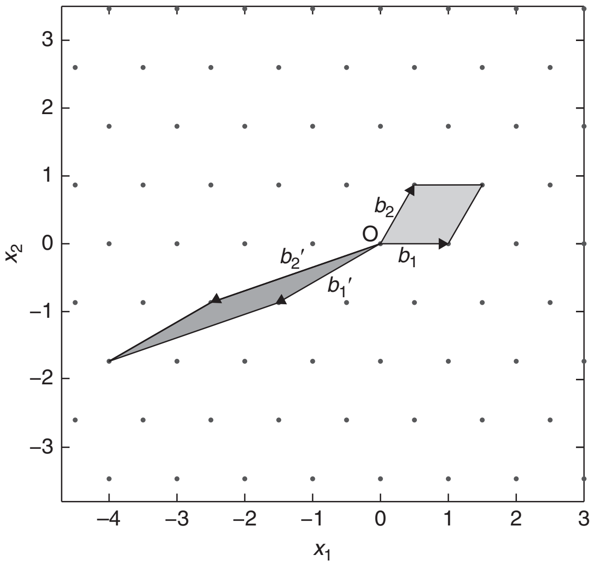

Figure 2. A lattice, with a reduced, and a nonreduced basis.

Given $b_1, \dots, b_r \in \mathbb{R}^n$, the vectors $b_1^*, \dots, b_r^*$ form their *Gram–Schmidt orthogonalization*, if

$$b_1^* = b_1 \tag{9}$$

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^* \, (i = 2, \dots, r), \tag{10}$$

with

$$\mu_{ij} = \langle b_i, b_j^* \rangle / \|b_j^*\|^2 \quad (i = 2, \dots, r; j = 1, \dots, i-1). \tag{11}$$

In other words, b_i^* is the projection of b_i onto the orthogonal complement of the linear span of $b_1, \dots, b_{i-1}$. Given $b_1, \dots, b_r$, their Gram–Schmidt vectors are by definition orthogonal, and will be used as reference vectors to express *near* orthogonality of the b_i.

Convention. Given $b_1, \dots, b_r$, we will write $b_1^*, \dots, b_r^*$ for their Gram–Schmidt vectors, and μ_{ij} for the corresponding coefficients without explicitly saying.

Perhaps the best-known reducedness concept in lattices is LLL reducedness. These bases were introduced by Lenstra and colleagues in their seminal 1982 article [14]. They are polynomial time computable when the lattice is generated by rational vectors, and they found uses in numerous contexts other than integer programming, for instance, in number theory and cryptography. For efficient implementations we refer the reader to the LiDia library [15] at the University of Darmstadt, and the NTL library of Victor Shoup [16].

We use Schrijver's definition [13], which is less restrictive than the classical one [14].

Definition 1. We say that $b_1, \dots, b_r$ is LLL reduced, if

1. $|\mu_{ij}| \leq 1/2 \ (i = 2, \dots, r; j = 1, \dots, i-1)$.
2. $\|b_i^*\| \leq \sqrt{2}\|b_{i+1}^*\| \ \ (i = 1, \dots, r-1)$.

We remark that an LLL reduced basis remains computable in polynomial time, if we replace the factor of $\sqrt{2}$ with $\sqrt{4/3 + \epsilon}$ for an arbitrarily small positive ϵ : the running time is polynomial in the size of the basis and $1/\epsilon$.

Both conditions in Definition 1 naturally correspond to "near orthogonality," since in an orthogonal basis all μ_{ij} would be zero, and condition (2) could be achieved by a simple rearranging, with the factor $\sqrt{2}$ replaced by 1. On the other hand, it is easy to construct examples of bases having near parallel vectors that violate conditions 1 and 2 by an arbitrary amount.

Exercise 2. Verify that the lattice in Example 2 does not have an orthogonal basis. Show that b_1 and b_2 form an LLL reduced basis, but b_1' and b_2' do not.

From the two basic properties of LLL reduced bases, one can derive several other inequalities that show the shortness (which is less straightforward from conditions (1) and (2) of Definition 1), and near orthogonality of the basis vectors. For details we refer to Lenstra *et al.* [14], and we recall an inequality that will be used in the section titled "Reformulation Methods".

Proposition 1. *Let $b_1, \dots, b_r$ be an LLL reduced basis of the lattice L, and $x_1, \dots, x_t$ be arbitrary linearly independent vectors in L. Then*

$$\max\left\{\|b_1\|, \dots, \|b_t\|\right\} \leq 2^{(r-1)/2} \max\left\{\|x_1\|, \dots, \|x_t\|\right\}. \tag{12}$$

We remark, that the performance of LLL reduction in practice is much better than predicted by the estimate (12), especially when t is small.

Korkin–Zolotarev (KZ) reduced bases were introduced by Kannan [2,3] as a key ingredient in his integer programming algorithm, and he also showed that they are computable in polynomial time, when the dimension is fixed. It is currently not known, whether they are computable in polynomial time for varying r. Surprisingly, it turned out that KZ reduced bases have been described previously already in the 19th century [17]; however, no algorithms were known to compute them until Kannan's result. The LiDia [15] and NTL libraries [16] also contain efficient subroutines to compute KZ reduced bases.

Definition 2. We say that $b_1, \dots, b_r$ is a KZ reduced basis, if

1. b_1 is a shortest vector of $\mathbb{L}(b_1, \dots, b_r)$;
2. $|\mu_{ij}| \leq 1/2$ $(i = 2, \dots, r; j = 1, \dots, i-1)$; and
3. letting $b'_2, \dots, b'_r$ be the projection of $b_2, \dots, b_r$ on the orthogonal complement of the line spanned by b_1, they form a KZ reduced basis.

While KZ reduced bases are harder to compute than LLL reduced ones, they have stronger properties. We recall two inequalities that we will use later.

Proposition 2. *Let $b_1, \dots, b_r$ be a KZ reduced basis of the lattice L, and $x_1, \dots, x_t$ arbitrary linearly independent vectors in L. Then*

$$\begin{aligned} &\max\left\{\|b_1\|, \dots, \|b_t\|\right\} \\ &\qquad \leq \frac{\sqrt{t+3}}{2} \max\left\{\|x_1\|, \dots, \|x_t\|\right\}. \end{aligned} \tag{13}$$

Furthermore, for all $j \leq r$

$$\|b_j^*\| \leq \sqrt{r-j+1}\left(\|b_j^*\| \dots \|b_r^*\|\right)^{1/(r-j+1)} \tag{14}$$

holds.

The inequality (13) is due to Lagarias *et al*. [18]. To put into context, we can compare the sublinear factor in it with the exponential factor in (12).

Inequality (14) follows from the definition of KZ reducedness, and for the proof we refer to Kannan [3]. Let us note that multiplying the inequalities $\|b_j^*\| \leq \sqrt{2}^i \|b_{j+i}^*\|$ for $i = 0, \dots, r-j$ in an LLL reduced basis would give a similar inequality with a factor of $2^{(r-j)/4}$ only (or with a factor $(4/3 + \epsilon)^{(r-j)/4}$, if we use a strengthened definition of LLL reducedness); so a KZ reduced basis improves inequality (2) in Definition 1 in an aggregate sense. Exercise 3 shows that the improvement also holds for every i.

Exercise 3. Show that if $b_1, \dots, b_r$ is a KZ reduced basis, then

$$\|b_i^*\| \leq \sqrt{4/3}\|b_{i+1}^*\| \quad (i = 1, \dots, r-1)$$

holds; so a KZ reduced basis is also LLL reduced.

THE GEOMETRY OF LATTICES AND CONVEX SETS

Proposition 3 concerns the solvability of $(\mathrm{IP}_{B,Q})$, and we will use it in the analysis of both the classical, and the reformulation methods. It asserts that if Q is "large" with respect to the norms of the columns of B, then $(\mathrm{IP}_{B,Q})$ is feasible, and if it is "small," then one can reduce $(\mathrm{IP}_{B,Q})$ to a "small" number of $r-1$ dimensional subproblems.

Proposition 3. *Denote the columns of B by $b_1, \dots, b_r$. Then*

1. *if Q is contained in the linear span of $\{b_1, \dots, b_r\}$, and*
$$r(Q) \geq \frac{1}{2}\sqrt{\|b_1^*\|^2 + \dots + \|b_r^*\|^2}, \tag{15}$$
then $(\mathrm{IP}_{B,Q})$ is feasible.
2. *If $(\mathrm{IP}_{B,Q})$ is feasible, then y_r must belong to an interval of length at most*
$$\left\lfloor \frac{2R(Q)}{\|b_r^*\|} \right\rfloor + 1; \tag{16}$$
hence $(\mathrm{IP}_{B,Q})$ can be reduced to at most this many $r-1$-dimensional IP feasibility problems.

For a proof of statement 1 of Proposition 3, we refer to Lenstra [1] (Lemma on p. 540). In fact, Lenstra only states the bound with b_i in place of the b_i^*, but his proof does imply the stronger statement. As motivation, consider a lattice in $\mathbb{R}^2$ spanned by orthogonal vectors, that is, $b_i^* = b_i$ for $i = 1, 2$. The point on the plane, which is farthest from the lattice points is at the intersection of diagonals of

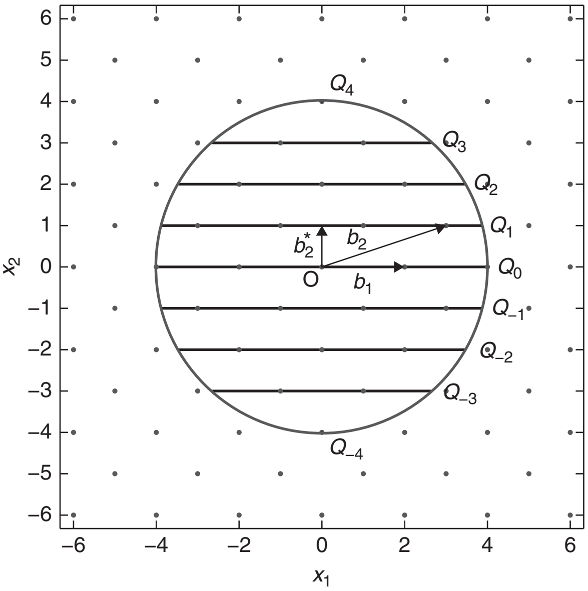

Figure 3. Illustration of statement 2 in Proposition 3.

a grid rectangle, with distance equal to the right-hand side of (15). Thus, a ball around any point in space with this radius will contain a lattice point.

Statement 2 of Proposition 3 is also essentially due to Lenstra, though it is not stated precisely in this form. A geometric proof can be found in Lenstra [1] (p. 541). For motivation, consider Fig. 3. The problem of finding a lattice point in Q is reduced to finding such a point in one of the Q_i, and the distance between the lines containing the Q_i is $\|b_2^*\|$, leading to the desired upper bound. Note that $\|b_2^*\|$ in this example is considerably smaller than $\|b_2\|$.

We give an algebraic proof of statement 2 of Proposition 3. Writing

$$B^{-1}Q = \{y \mid By \in Q\} \tag{17}$$

we show

$$\mathrm{width}(e_r, B^{-1}Q) \le \frac{2R(Q)}{\|b_r^*\|}. \tag{18}$$

Let $y_{r,1}$ and $y_{r,2}$ denote the maximum and the minimum of y_r over $B^{-1}Q$. Writing $\overline{B}$ for the matrix composed of the first $r-1$ columns of B, and b_r for the last column, it holds that there are $y_1, y_2 \in \mathbb{R}^{r-1}$ such that $\overline{B}y_1 + b_r y_{r,1}$ and $\overline{B}y_2 + b_r y_{r,2}$ are in Q. So,

$$\begin{aligned} 2R(Q) &\ge \left\|(\overline{B}y_1 + b_r y_{r,1}) - (\overline{B}y_2 + b_r y_{r,2})\right\| \\ &= \left\|\overline{B}(y_1 - y_2) + b_r(y_{r,1} - y_{r,2})\right\| \\ &\ge \left\|b_r^*\right\| |y_{r,1} - y_{r,2}| \\ &= \left\|b_r^*\right\| \mathrm{width}(e_r, B^{-1}Q) \end{aligned}$$

holds, as required.

Remark 1. If y_r is fixed to an integer k in $(\mathrm{IP}_{B,Q})$, then the corresponding lower-dimensional subproblem is

$$\text{Find } \overline{y} \in \mathbb{Z}^{r-1} : \overline{B}\overline{y} + kb_r \in Q \cap \{y \mid y_r = k\}. \tag{19}$$

Corollary 1. *When in* $(\mathrm{IP}_{B,Q})$ *we branch on* $y_r, y_{r-1}, \ldots, y_j$ *in this order, the number of resulting subproblems on the level of* y_j *is at most*

$$\prod_{i=j}^{r} \left(\left\lfloor \frac{2R(Q)}{\|b_i^*\|} \right\rfloor + 1 \right). \tag{20}$$

Proof. The formula (19) shows the form of the resulting subproblems, after we branched on y_r. Using part (2) in Proposition 3 implies that branching on y_{r-1} will create at most

$$\left\lfloor \frac{2R(Q)}{\|b^*_{r-1}\|} \right\rfloor + 1 \tag{21}$$

subproblems from each of these, and so on, and this completes the proof.

LENSTRA'S AND KANNAN'S ALGORITHMS

Lenstra's Algorithm

Lenstra's paper [1] first disposes with two technicalities. Assuming that P is described as $P = \{x \,|\, Ax \le b\}$, with A and b integral, and their components bounded by a in absolute value, Lenstra shows that when (IP) is feasible, it has a solution $\bar{x}$ with $\max_i |\bar{x}_i| \le (n+1)n^{n/2}a^n$. So we can assume that P is bounded, and Lenstra shows that we can also assume that it is full dimensional.

Next we "round" P by computing in polynomial time an invertible transformation τ, so that

$$R(\tau P)/r(\tau P) \le c_n := n(n+1) \tag{22}$$

will hold, that is, τP appears relatively "spherical."

Example 1 (continued). Using Lenstra's algorithm we computed τP, where P is the polyhedron in Example 1. With τ having a transformation matrix

$$\begin{pmatrix} 0.3326 & 0.3101 \\ 0 & 0.0164 \end{pmatrix}, \tag{23}$$

the rounded polyhedron is

$$\begin{aligned} 460 \le\ & 153.3333x_1 + 88.5270x_2 \le 489 \\ 0 \le\ & \ \ 3.0065x_1 - 56.8034x_2 \ \ \le 10 \\ 0 \le\ & \qquad\quad 60.9285x_2 \qquad\quad\ \le 10, \end{aligned} \tag{24}$$

shown in Fig. 4 with the points of the lattice $\tau\mathbb{Z}^2$.

Now (IP) is transformed into trying to find $x \in \tau\mathbb{Z}^n \cap \tau P$. After choosing an LLL reduced basis B for the lattice $\tau\mathbb{Z}^n$, the task becomes

$$\text{Find } y \in \mathbb{Z}^n : By \in \tau P. \tag{IP$_{B,\tau P}$}$$

Let us write $R := R(\tau P), r := r(\tau P)$, and let $b_1, \ldots, b_n$ denote the columns of B. For better intuition, we first describe the algorithm assuming $R = r$ (i.e., that τP *is* a ball), and that the b_i^* all have the same norm.

By Proposition 3 if

$$r \ge \sqrt{n}\|b_n^*\|/2, \tag{25}$$

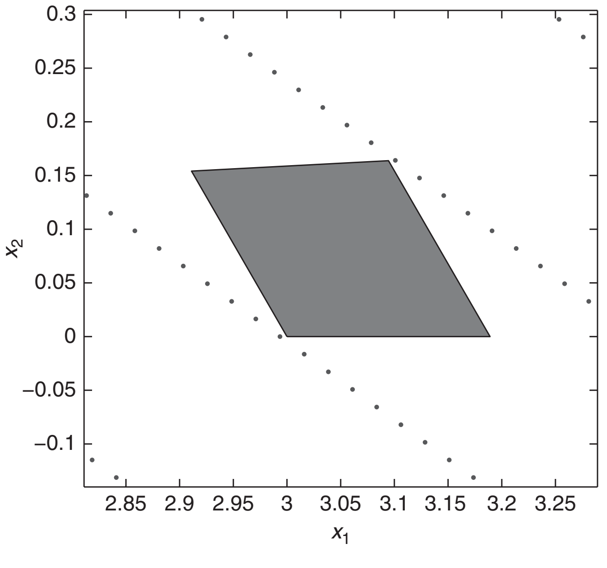

Figure 4. The polyhedron of Example 1 after Lenstra's rounding, and the points of τF^2.

then $(\mathrm{IP}_{B,\tau P})$ is feasible, and if (25) fails, $(\mathrm{IP}_{B,\tau P})$ can be reduced to at most

$$\left\lfloor \frac{2r}{\|b_n^*\|} \right\rfloor + 1 \leq \sqrt{n} + 1$$

$n-1$-dimensional subproblems. On the subproblems we apply the algorithm recursively, by making the underlying polyhedra full dimensional, rounded, and finding a new LLL reduced basis for each one. If we denote by $f(n)$ the number of polyhedra that Lenstra's algorithm must examine, we obtain

$$f(n) \leq (\sqrt{n}+1)f(n-1); \qquad (26)$$

thus, $f(n) = O(n^{n/2})$. The number of arithmetic operations performed on each polyhedron is polynomial, so the overall complexity is polynomial, when n is fixed.

The roundedness of τP, and the reducedness of B make sure that our unrealistic assumption is not far from the truth. Let j be the index such that $\|b_j^*\|$ is maximal, and consider the two cases:

Case 1.

$$r \geq \frac{1}{2}\sqrt{n}\|b_j^*\|.$$

Then the first statement in Proposition 3 shows that $(\mathrm{IP}_{B,\tau P})$ is feasible.

Case 2.

$$r < \frac{1}{2}\sqrt{n}\|b_j^*\|.$$

Then

$$R < \frac{1}{2}c_n\sqrt{n}\|b_j^*\| < \frac{1}{2}c_n\sqrt{n}\|b_n^*\|2^{(n-j)/2},$$

with the first inequality from (22), and the second from the LLL reducedness of B. Then Proposition 3 implies that branching on y_n in $(\mathrm{IP}_{B,\tau P})$ creates at most

$$\left\lfloor \frac{2R}{\|b_n^*\|} \right\rfloor + 1 = O(2^n) \qquad (27)$$

subproblems.

Denoting by $f_L(n)$ the number of polyhedra that must be examined by Lenstra's algorithm, we have

$$f_L(n) = O(2^n)f_L(n-1) = O(2^{n^2}), \qquad (28)$$

and the previous argument shows polynomiality of the method for fixed n.

Exercise 4. Verify that if P is the polyhedron in Example 1, and τ is given in (23), then τP indeed has the description given in (24), and $R(\tau P)/r(\tau P) \leq 3.82$.

In contrast, show that the long and thin nature of P is also shown by the poor ratio of $R(P)$ and $r(P)$, namely $R(P)/r(P) \geq 33$.

Kannan's Algorithm

In Kannan's algorithm the setup of making P bounded, full dimensional, and rounded is the same as in Lenstra's method; then a reduced basis for the lattice $\tau\mathbb{Z}^n$ is found. Hence (IP) is again transformed into $(\mathrm{IP}_{B,\tau P})$, and the handling of Case 1 is identical. However, Kannan uses a Korkin–Zolotarev reduced basis for $\mathbb{L}(B)$ in $(\mathrm{IP}_{B,\tau P})$, and again letting j to be the index for which the norm of the corresponding Gram–Schmidt vector is maximal, we enumerate all possible values for $y_j, \ldots, y_n$. So, if $j = n$, Kannan's algorithm behaves like Lenstra's, and if $j = 1$, it does complete enumeration.

The complexity analysis of Kannan's algorithm is more technical than Lenstra's, so we only outline a proof showing that it needs to look at only $O(n^{3n})$ polyhedra, as opposed to Lenstra's $O(2^{n^2})$. We first describe Case 2 in detail.

Case 2.

$$r < \frac{1}{2}\sqrt{n}\|b_j^*\|,$$

that is

$$R < \frac{1}{2}c_n\sqrt{n}\|b_j^*\|.$$

Corollary 1 implies that branching on $y_n, \ldots, y_j$ creates at most

$$\prod_{i=j}^{n}\left(\left\lfloor \frac{2R}{\|b_i^*\|} \right\rfloor + 1\right).$$

subproblems. Using the fact that b_j^* has the largest norm among the Gram–Schmidt vectors, simple manipulation shows that this upper bound is at most

$$(c_n\sqrt{n}+1)^{n-j+1}\frac{\|b_j^*\|^{n-j+1}}{\|b_j^*\|\dots\|b_n^*\|}.$$

We now use the KZ reducedness of the b_i, and plug the estimate (14) into the latter expression. Hence the number of subproblems is at most

$$\left(c_n\sqrt{n}+1\right)^{n-j+1}\left(\sqrt{n-j+1}\right)^{n-j+1},$$

which is $O(n^{3(n-j+1)})$. So, denoting by $f_K(n)$ the number of polyhedra that must be examined by Kannan's algorithm,

$$f_K(n)=O(n^{3(n-j+1)})f_K(j-1)=O(n^{3n})$$

holds.

REFORMULATION METHODS

In this section we describe the rangespace and nullspace reformulation methods. It will be convenient to assume that our feasibility problem is given as

$$\begin{pmatrix}\ell_1\\ \ell_2\end{pmatrix}\le\begin{pmatrix}A\\ I\end{pmatrix}x\le\begin{pmatrix}w_1\\ w_2\end{pmatrix}\qquad x\in\mathbb{Z}^n, \tag{29}$$

where A is an integral m by n matrix. The rangespace reformulation of (29) is

$$\begin{pmatrix}\ell_1\\ \ell_2\end{pmatrix}\le\begin{pmatrix}A\\ I\end{pmatrix}Uy\le\begin{pmatrix}w_1\\ w_2\end{pmatrix}\qquad y\in\mathbb{Z}^n, \tag{30}$$

where U is a unimodular matrix computed to make the columns of the constraint matrix a reduced basis of the generated lattice. If y is feasible for (30), then Uy is feasible for (29), and if x is feasible for (29), then $U^{-1}x$ is feasible for (30), since U^{-1} is integral. This shows the equivalence of the two problems.

The nullspace reformulation is applicable when $w_1=\ell_1$. Assuming that the rows of A are linearly independent, it is

$$\ell_2-x_0\ \le\ By\ \le w_2-x_0\qquad y\in\mathbb{Z}^{n-m}, \tag{31}$$

where $x_0\in\mathbb{Z}^n$ satisfies $Ax_0=\ell_1$, and B is a reduced basis of $\mathbb{N}(A)$ (recall the definition of the null-lattice from (5)). Since any integral solution of the system $Ax=\ell_1$ can be written as the sum of x_0, and an element of $\mathbb{N}(A)$, the two formulations are equivalent [13].

For brevity, if the columns of the constraint matrix in (30) form an LLL reduced basis of the generated lattice, we will call (30) the *LLL rangespace reformulation of* (29), and we will similarly talk about the LLL nullspace reformulation, KZ rangespace reformulation, and so on.

Remark 2. In its simplest form, the rangespace reformulation method transforms the feasibility problem

$$Ax\ \#\ b, x\in\mathbb{Z}^n$$

into

$$AUy\ \#\ b, y\in\mathbb{Z}^n,$$

where U is a unimodular matrix, AU is a reduced basis of the lattice $\mathbb{L}(A)$, and # denotes an arbitrary mixture of equality and inequality constraints. For the analyses that we present, however, it is convenient to work with a problem in the form of (29).

Also, one can solve an integer programming optimization problem by reducing it to a sequence of feasibility problems, on which a suitable reformulation can be then applied. The simplest direct approach is transforming

$$\max\left\{cx\mid Ax\ \#\ b, x\in\mathbb{Z}^n\right\}$$

into

$$\max\left\{cUy\mid AUy\ \#\ b, y\in\mathbb{Z}^n\right\},$$

where again # is a mixture of equality and inequality constraints, U is a unimodular matrix, and

$$\begin{pmatrix} c \\ A \end{pmatrix} U$$

is a reduced basis of the generated lattice.

Let us note how the reformulated problems fit in the framework of ($\mathrm{IP}_{B,Q}$) with the polyhedron Q being simply a box, and B constructed directly from the constraint matrix of (29). This is in contrast with Lenstra's and Kannan's algorithms, where Q is a transformed version of P, and B is a reduced basis of the transformation matrix.

Decomposable Knapsack Problems and Their Reformulations

In this section we are interested in feasibility problems of the type

$$\begin{pmatrix} \ell_1 \\ \ell_2 \end{pmatrix} \leq \begin{pmatrix} a \\ I \end{pmatrix} x \leq \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \qquad x \in \mathbb{Z}^n, \tag{32}$$

with a decomposing as

$$a = \lambda p + r, \tag{33}$$

where a, p, and r are integral row vectors, the components of p relatively prime, and λ is an integer. We will call these *decomposable knapsack problems (DKPs),* and note that this is not a formal definition, since any knapsack problem with a constraint vector a can be written in this form with $p = a, r = 0$, and $\lambda = 1$. However, in all interesting cases λ will be at least 2, and in most cases relatively large with respect to $\|p\|$ and $\|r\|$.

There are a wide variety of instances, which fit into this framework, with three interesting properties, that we repeat from the introduction:

(i) they are difficult for ordinary B&B;
(ii) their infeasibility is proven by branching on px; and
(iii) when λ is sufficiently large, branching on the last variable in the reformulations has the same effect as branching on px in the original problem.

Example 1 (continued). This instance fits the mold of (32) with the constraint vector, and its decomposition given by

$$a = (51, 49), \quad p = (1, 1), \qquad r = (1, -1), \quad \lambda = 50. \tag{34}$$

The definition of the rest of the data (of ℓ_1, ℓ_2, w_1, and w_2) is obvious.

With P denoting the underlying polyhedron, we have

$$\begin{aligned} \max\{px \mid x \in P\} &= 489/49 = 9.9796, \\ \min\{px \mid x \in P\} &= 460/51 = 9.0196; \end{aligned} \tag{35}$$

so, branching on px proves the infeasibility of this instance, and it is straightforward to check that the same holds for the generalization of Exercise 1.

The next example is a simplification of Jeroslow's classic problem [7], which established that ordinary B&B may take exponential time even on a trivial problem.

Example 3. Let n be a positive, odd integer. The problem

$$\begin{aligned} 2\sum_{i=1}^{n} x_i &= n \\ 0 \leq x &\leq e \\ x &\in \mathbb{Z}^n \end{aligned} \tag{36}$$

is infeasible, and the infeasibility is proven by branching on $\sum_{i=1}^{n} x_i$. At the same time, ordinary B&B needs to enumerate at least $2^{(n-1)/2}$ nodes to do the same. For a proof of the latter fact we refer to Jeroslow's [7] or Krishnamoorthy and Pataki's paper [5] (p. 247).

This instance also fits into the framework of (32) with

$$a = 2e, \; p = e, \; r = 0, \; \lambda = 2, \tag{37}$$

and $\ell_1 = w_1 = n$.

Another family of instances with similar behavior is related to the famous *Frobenius problem.* For a positive integral vector $a =$

$(a_1, \ldots, a_n)$, the *Frobenius number* Frob(a) is the largest integer for which we cannot make a change using the denominations $a_1, \ldots, a_n$; in other words, the largest β for which the integer programming problem

$$\begin{aligned} ax &= \beta \\ x &\geq 0 \\ x &\in \mathbb{Z}^n, \end{aligned} \tag{38}$$

is infeasible. The *Frobenius problem* is computing Frob(a) : for a recent review we refer to a book by Ramirez Alfonsin [19].

The Frobenius problem gives rise to interesting, and difficult integer programming instances. Cornuéjols *et al.* [20] studied knapsack problems with a decomposable structure, and showed how to compute the Frobenius number by solving a sequence of integer programming problems using a test set approach.

Aardal and Lenstra [8,9] considered instances of the form (38), with a decomposing as in (33), and β a positive integer. Following the arguments in Krishnamoorthy and Pataki [5], we give a simplified description of the instances they constructed, and in addition show that they fit Property (ii).

Example 4. Consider a knapsack feasibility problem (38) with a decomposing as in (33), and β a positive integer. We assume that p is componentwise positive, it is not a multiple of r, and

$$q_1 := r_1/p_1 \leq \cdots \leq q_n := r_n/p_n. \tag{39}$$

Proposition 4. *Define*

$$\begin{aligned} k &= \lfloor (\lambda + q_1 - 1)/(q_n - q_1) \rfloor - 1, \\ \beta &= \lceil (\lambda + q_n)k \rceil, \end{aligned} \tag{40}$$

and assume that λ is large enough so k and β are both positive. Then the infeasibility of (38) is proven by branching on px.

Proof. Let us consider the interval

$$I = ((\lambda + q_n)k, (\lambda + q_1)(k+1)), \tag{41}$$

and note that I has length strictly larger than one by the choice of k. Next, let us denote by P the polyhedron of the LP relaxation of (38). Since $a_i = \lambda p_i + r_i$, the ordering in (39) implies

$$a_1/p_1 \geq \cdots \geq a_n/p_n, \tag{42}$$

therefore

$$\begin{aligned} \max\{px \mid x \in P\} &= p_1\beta/a_1, \\ \min\{px \mid x \in P\} &= p_n\beta/a_n, \end{aligned} \tag{43}$$

hold. So iwidth$(p, P) = 0$ holds, when the above maximum and minimum are in the interval $(k, k+1)$. A simple calculation shows that this happens exactly when $\beta \in I$, which is true because β is the ceiling of the lower end point of I, and $|I| > 1$.

Since (38) is infeasible with a large right-hand side, it is difficult for ordinary B&B, and it is also difficult when the right-hand side is chosen to be Frob(a).

The following exercises review the recipe from Krishnamoorthy and Pataki [5], and its application to construct Avis's instance from Chvátal [21]:

Exercise 5. Let p and r be arbitrary integral vectors, ℓ_2 and w_2 bound vectors in (32), and k an integer with $0 < k \leq pw_2$. Prove that if we find λ, ℓ_1, and w_1 to satisfy $\ell_1 \leq w_1$, and

$$\begin{aligned} &\max\{rx \mid px \leq k, \ell_2 \leq x \leq w_2\} + k\lambda < \ell_1, \\ &\qquad \text{and} \\ &\min\{rx \mid px \geq k+1, \ell_2 \leq x \leq w_2\} \\ &\qquad + (k+1)\lambda > w_1, \end{aligned} \tag{44}$$

then the infeasibility of (32) is proven by branching on px.

Show that the instances of Examples 1, 3, and 4 can be obtained this way, with $k = 9$, $k = (n-1)/2$, and k defined in (40), respectively.

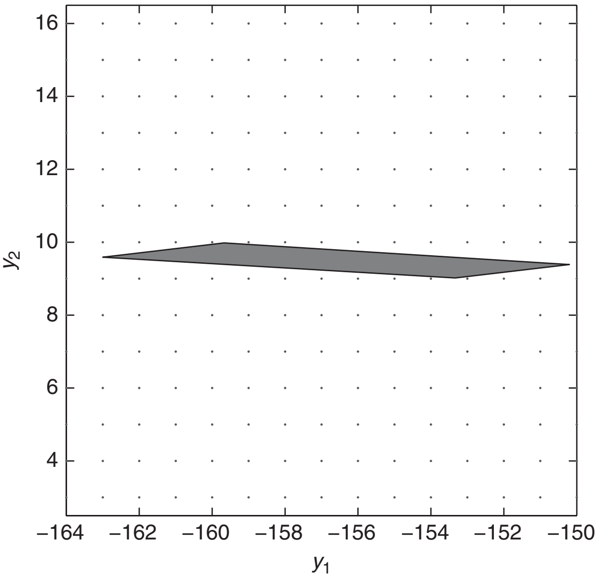

Figure 5. Example 1 after applying the LLL rangespace reformulation.

For a proof of why the above recipe creates hard integer programs, and how the difficulty depends on p, r, λ, and k, we refer to Krishnamoorthy and Pataki [5].

Exercise 6. Using the result of Exercise 5, show that the subset sum problem

$$\begin{aligned} ax &= \beta \\ x &\in \{0,1\}^n, \end{aligned} \tag{45}$$

where n is odd,

$$\begin{aligned} a &= (n(n+1)+1, \ldots, n(n+1)+n), \\ \beta &= \left\lfloor \sum_{i=1}^{n} a_i/2 \right\rfloor \end{aligned} \tag{46}$$

is infeasible. This instance was proposed by Avis [21], and it is known that ordinary B&B needs at least $2^{(n-1)/2}$ nodes to prove its infeasibility.

Example 1 (continued). To motivate Theorem 1, the main result presented in this section, we computed the LLL rangespace reformulation of Example 1. With

$$U = \begin{pmatrix} -1 & -16 \\ 1 & 17 \end{pmatrix},$$

it is

$$\begin{aligned} 460 \leq\ & 2y_1 + 17y_2 \leq 489 \\ 0 \leq\ & -y_1 - 16y_2 \leq 10 \\ 0 \leq\ & y_1 + 17y_2 \leq 10 \\ & y_1, y_2 \in \mathbb{Z}, \end{aligned} \tag{47}$$

shown in Fig. 5. It is interesting to note that the underlying polyhedron is still long and thin, unlike the polyhedron in Fig. 4 produced by the τ transformation in Lenstra's and Kannan's methods, but now infeasibility is proven by branching on y_2.

Theorem 1. *Denote by P the polyhedron of (32), and by Q the polyhedron of its LLL rangespace reformulation. If*

$$\lambda > 2^{(n-1)/2}(\|r\|+1)^2\|p\|, \tag{48}$$

then

$$\begin{aligned} \text{width}(e_n, Q) &= \text{width}(p, P), \text{ and} \\ \text{iwidth}(e_n, Q) &= \text{iwidth}(p, P). \end{aligned} \tag{49}$$

Sketch of proof Let us write

$$A = \begin{pmatrix} a \\ I \end{pmatrix}. \tag{50}$$

First we show that $\mathbb{L}(A)$ has at least $n-1$ vectors with norm bounded by $(\|r\|+1)\|p\|$.

Indeed, there are $n-1$ vectors in $\mathbb{N}(p)$ with norm at most $\|p\|$, and if w is such a vector, then

$$\|Aw\| \le (\|r\|+1)\|p\|. \tag{51}$$

Next, let U be the unimodular matrix in the LLL rangespace reformulation of (32). Since AU is an LLL reduced basis of $\mathbb{L}(A)$, Proposition 1 shows that its first $n-1$ columns have norm of at most

$$2^{(n-1)/2}(\|r\|+1)\|p\|. \tag{52}$$

The integrality of pU, and the choice of λ implies that the first $n-1$ components of pU are 0, otherwise the corresponding column of AU would have norm larger than as given in (52). For the details we refer to Krishnamoorthy and Pataki [5] (p. 262).

Hence $pU = \delta e_n$ holds for some integer δ, that is, $p = \delta e_n U^{-1}$, and since the components of p are relatively prime, $\delta = \pm 1$.

Finally, the definitions of P and Q imply

$$\max\{px \mid x \in P\} = \max\{pUy \mid y \in Q\}, \tag{53}$$

and the same holds for the minimum. Using $pU = \pm e_n$ with (53), and the definition of width and integer width prove (49).

Remark 3. One can see that an analogous result holds for the KZ rangespace reformulation, even if we replace the lower bound (48) with the weaker

$$\lambda > \sqrt{(n+3)/2}(\|r\|+1)^2\|p\|. \tag{54}$$

A variant of Theorem 1 about the nullspace reformulation is proven in Krishnamoorthy and Pataki [5].

Remark 4. Another decomposable knapsack instance, similar to Avis's, was described by Todd in Chvátal's 1980 paper [21]. It is interesting that Jeroslow, Avis, and Todd proved that their instances fit property (i), but they did not mention that they fit property (ii) as well.

Aardal and Lenstra [8,9] showed that denoting by b_{n-1} the last column of the constraint matrix in the nullspace reformulation

$$\|b_{n-1}\| \ge \|b^*_{n-1}\| = \Omega(\lambda)$$

holds, and they argued that $\|b_{n-1}\|$ being long implies that branching on y_{n-1} will generate a small number of subproblems.

Krishnamoorthy and Pataki [5] pointed out a gap in the proof of $\|b^*_{n-1}\| = \Omega(\lambda)$, and constructed an example of a polyhedron $Q = \{y \in \mathbb{R}^r \mid \ell \le By \le w\}$, where the columns form an LLL reduced basis of $\mathbb{L}(B)$, but branching on y_r creates $c^r\|b_r\|$ subproblems for some $c > 1$. Furthermore, they proved that the instance of Example 4 fits property (ii).

Analyzing the Reformulation Methods without Assuming Structure

Here we describe an analysis of the reformulation methods based on the paper of Pataki *et al.* [10], without assuming any structure on the matrix A in (29). Interestingly, we will find that ordinary B&B solves the reformulation of the *majority* of the instances *without any branching*. We explain the connection with solving low-density subset sum problems after the proof.

We assume $n \ge 5$, and when a statement is true for all, but at most a fraction of $1/2^n$ of the elements of a set S, we say that it is true for *almost all* elements. For positive integers m, n, and M we denote by $G_{m,n}(M)$ the set of matrices with m rows and n columns, and the entries from $\{1, \ldots, M\}$, and by $G'_{m,n}(M)$ the subset of $G_{m,n}(M)$ consisting of matrices with linearly independent rows.

We use a version of ordinary B&B that branches on the variables in reverse order, and call this algorithm *reverse B&B*. If B&B generates at most one node at each level of the tree, we say that it solves an integer feasibility problem *at the root node*. When the system $Ax = \ell_1$ does not have an integral solution, the nullspace reformulation does not exist: for simplicity we still say that in this case the reformulated instance is solved at the root node.

Theorem 2. *If $M \geq (2^{(n+4)/2}\|(w_1; w_2) - (\ell_1; \ell_2)\|)^{n/m+1}$, then for almost all $A \in G_{m,n}(M)$ reverse B&B solves the LLL rangespace reformulation of (29) at the root node.*

Also, if $M \geq (2^{(n-m+4)/2}\|w_2 - \ell_2\|)^{n/m}$, then for almost all $A \in G'_{m,n}(M)$ reverse B&B solves the LLL nullspace reformulation of (29) at the root node.

Proof Sketch We outline a proof of the first statement, and refer the reader to Pataki *et al*. [10] for details, and the proof of the second. For convenience, we shall write $(A; I)$ for the matrix obtained by stacking A on top of I, and the meaning of $(\ell_1; \ell_2)$ and $(w_1; w_2)$ will be analogous.

Let U be the matrix such that the columns of $(A; I)U$ form an LLL reduced basis of the generated lattice. We first use Corollary 3 with $B = (A; I)U$, and $Q = \{y \mid (\ell_1; \ell_2) \leq y \leq (w_1; w_2)\}$ to find that when reverse B&B is applied to (30), the number of B&B nodes on the level of y_j is at most

$$\prod_{i=j}^{n}\left(\left\lfloor \frac{\|(w_1; w_2) - (\ell_1; \ell_2)\|}{\|b_i^*\|} \right\rfloor + 1\right),$$

where $b_1^*, \ldots, b_n^*$ form the Gram–Schmidt orthogonalization of the columns of $(A; I)U$. Hence if

$$\|b_i^*\| > \|(w_1; w_2) - (\ell_1; \ell_2)\| \quad \forall i = 1, \ldots, n, \tag{55}$$

then the problem is solved at the root node.

The definition of LLL reducedness implies

$$\begin{aligned} \|b_i^*\| &\geq \frac{1}{2^{(i-1)/2}}\|b_1\| \\ &\geq \frac{1}{2^{(i-1)/2}}\lambda_1(\mathbb{L}(A; I)), \end{aligned} \tag{56}$$

where $\lambda_1(\mathbb{L}(A; I))$ denotes the length of the shortest nonzero vector in $\mathbb{L}(A; I)$. So (55) holds, when

$$\lambda_1(\mathbb{L}(A; I)) > 2^{(n-1)/2}\|(w_1; w_2) - (\ell_1; \ell_2)\|. \tag{57}$$

Condition (57) does not hold for all A matrices, so let us call a matrix $A \in G_{m,n}(M)$ *bad,* when (57) fails. One can show that for $r > 0$ the shortest vector in $\mathbb{L}(A; I)$ is strictly longer than r for all, but at most a fraction of

$$\frac{(2\lfloor r \rfloor + 1)^{n+m}}{M^m} \tag{58}$$

matrices in $G_{m,n}(M)$. We refer the reader to Lemma 2.2 in Pataki *et al*. [10] for details.

Using this result, it follows that when M is as given in the first statement of the theorem, the fraction of bad matrices is at most $1/2^n$, and this completes the proof.

Remark 5. A stronger version of Theorem 2 is true, if we use a "more reduced" basis, in particular, a so-called *reciprocal KZ reduced basis*. For details, we refer to Pataki *et al*. [10].

There is an interesting connection with earlier work on subset sum problems, which we outline here. Furst and Kannan [22] based on Lagarias' and Odlyzko's [23] and Frieze's [24] work show that the subset sum problem

$$\begin{aligned} ax &= \beta \\ x &\in \{0,1\}^n, \end{aligned} \tag{59}$$

is solvable in polynomial time using a simple iterative method for almost all $a \in G_{1,n}(M)$, and all right-hand sides, when M is sufficiently large, and a reduced basis of $\mathbb{N}(a)$ is available. Their bound on M is $2^{n^2/2+2n}n^{3n/2}$, when the basis is LLL reduced, and $2^{(3/2)n\log n+5n}$, when it is reciprocal KZ reduced.

Subset sum problems with potentially such large coefficients find uses in cryptography. The vector a is a public key, x is a message, and the encoded message that the sender transmits is ax. The wide range of the coefficients of a makes sure that few right-hand sides among the integers in $\{1, \ldots, \sum_{i=1}^{n} a_i\}$ arise as ax for some $x \in \{0,1\}^n$, and it is rare for two distinct x vectors to map to the same ax. The results of Lagarias and Odlyzko [23], Frieze [24], and a later improvement by Coster *et al*. [25] show that using basis reduction the solution of (59) can be found with high probability, if it is known to exist. In other words, an intercepted message can be decoded for most of the a public keys. Furst and Kannan [22] go further by finding the solution if it exists, and a proof of infeasibility for instances that do not have a solution.

Our bounds obtained by letting $m = 1$ in Theorem 2 and its variant that uses reciprocal KZ reducedness are comparable to Furst and Kannan's bounds, when the size of M, that is, $\lceil \log(M + 1) \rceil$ is concerned. Precisely, the bound on the size of M is $O(n^2)$, when we use an LLL reduced basis, and $O(n \log n)$, when we use a reciprocal KZ reduced basis.

So these results generalize the solvability results of [22] from subset sum problems to bounded integer programs. It is also interesting that one can prove complexity results via branch-and-bound, an algorithm that has been considered inefficient from the theoretical point of view.

FURTHER READING AND COMPUTATIONAL TESTING

In this section we briefly review results on integer programming in fixed dimension, whose detailed treatment is beyond the scope of our survey, mention other surveys that the reader may find to be of interest, and review computational experience with lattice-based methods. We will write "polynomial time when the dimension is fixed" as *fd-polynomial time* for short.

The generalized basis reduction algorithm of Lovász and Scarf [26] also solves (IP) in fd-polynomial time. Instead of rounding the underlying polytope, like Lenstra's and Kannan's algorithms do, at its core is a subroutine that finds an integral vector p such that width(p, P) is relatively small, by solving a sequence of linear programs. This vector is then used for branching. Kannan [27] presented an algorithm to solve the Frobenius problem in fd-polynomial time.

An fd-polynomial time algorithm exists even to *count* the number of feasible solutions of (IP). The breakthrough algorithm to achieve this is due to Barvinok [28]. His algorithm was considerably simplified by Dyer and Kannan [29], and successfully implemented by De Loera *et al*. [30]. Koeppe [31] developed, and implemented a newer primal variant, which in many cases is also faster in practice.

Given $c \in \mathbb{Z}^n$, the integer optimization problem is finding a feasible solution of (IP), which maximizes cx. One can solve this problem by reducing it to a sequence of feasibility problems; however, it is interesting to study direct approaches, which are theoretically efficient. We refer to Eisenbrand [32] for a fast algorithm to solve this problem, under the assumption that the number of variables, and the number of constraints are both fixed. Computing the *integer programming gap* is the problem of finding the maximum difference between the optimal value of an integer programming problem, and its LP relaxation, as the right-hand side varies. An fd-polynomial time algorithm for this problem was developed by Hoşten and Sturmfels [33], assuming that the number of constraints is also fixed. Eisenbrand and Shmonin [34] described an fd-polynomial algorithm even when the number of constraints is allowed to vary.

Other reviews on the uses of basis reduction and integer programming that the reader may find useful are by Kannan [35], Aardal and Eisenbrand [36], and Eisenbrand [37]: the latter is also a tutorial, with accompanying exercises. A substantial part of the books of Schrijver [13], and Grötschel *et al*. [38] are also devoted to this subject.

There is surprisingly little experience with implementing Lenstra's algorithm. Gao and Zhang [39] described an implementation, and Mehrotra and Li [6] presented and implemented a nonrecursive variant, in which branching is done on a hyperplane in the original formulation. Cook *et al*. [40] reported a successful implementation of the generalized basis reduction method.

There is more computational evidence of the effectiveness of the reformulation methods.

Aardal *et al*. [4] successfully tested their reformulation on knapsack-type feasibility problems that arise from circuit design. Another application of lattice-based methods is on the *marketshare* problems of Cornuéjols and Dawande [41]. Suppose that a company supplies n retailers with m products, with retailer j receiving a_{ij} units of product i. The company has two divisions. We would like to assign each retailer to one of the divisions, so the retailers in each division receive approximately half of the total supply of each product.

Letting A be a matrix with the (i,j)th entry equal to a_{ij}, and $b \in \mathbb{Z}^m$ with the ith entry equal to

$$\left\lfloor \frac{1}{2} \sum_{j=1}^{n} a_{ij} \right\rfloor,$$

the problem can be formulated as

$$\begin{aligned} Ax &= b \\ x &\in \{0,1\}^n, \end{aligned} \tag{60}$$

where x_j is set to 1, if retailer j is assigned to division 1, and 0 otherwise.

Two variants of (60) have also been studied, which are especially interesting, when the original problem is infeasible. The first is an optimization version introduced in Cornuéjols and Dawande [41], which attempts to minimize $\|Ax - b\|_1$. The second is a relaxed version studied in Pataki *et al.* [10], namely,

$$\begin{aligned} b - e \;\leq\; Ax \;&\leq\; b \\ x \;&\in\; \{0,1\}^n. \end{aligned} \tag{61}$$

This formulation attempts to find a near equal market split (of course if $\sum_{j=1}^{n} a_{ij}$ is odd, then the ith constraints in (61) are as good as the ith constraint in (60)).

The marketshare problems are exceptionally difficult to solve by commercial integer programming software, and Cornuéjols and Dawande offered them as a challenge to the Integer Programming community. They generated the a_{ij} uniformly at random in the interval $\{1, \ldots, 100\}$, with the choice $n = 10(m-1)$. Aardal *et al.* [42] showed that by using the CPLEX 6.5 commercial Mixed Integer Programming (MIP) solver, the nullspace reformulations of 7 × 60 instances were solved in a reasonable amount of time, whereas the original formulations of even 5 × 60 instances could not be handled. Improved results were obtained for the optimization versions as well, and the authors also derived an approximation for the number of feasible solutions of (60) in terms of m and n. A generalization of the marketshare problem with a matrix variables, and two-sided constraints was studied by Louveaux and Wolsey [43].

A counterintuitive guess based on Theorem 2 is that the reformulations of the marketshare problems should become *easier* in practice, when the a_{ij} are drawn from $\{1, \ldots, M\}$, and M grows. This was confirmed by computational experiments by Pataki *et al.* [10]. For instance, the average number of nodes over 12 instances that needed to be enumerated by CPLEX 9 to solve the rangespace reformulation of 5 × 40 relaxed instances with $M = 100$ was 38865. However, the average number of nodes with $M = 10000$ was just 1976.

A computational study on the Frobenius instances given in Example 4 was carried out by Aardal and Lenstra [8]. Krishnamoorthy and Pataki [5] experimented with more general DKPs, both with feasible and infeasible instances, and using varying bounds. As expected, the reformulations were quite easy to solve, usually requiring less than a hundred nodes, even when λ was not large enough for Theorem 1 (or its version using KZ reducedness) to give theoretical guarantees.

Two other interesting observations were made in Krishnamoorthy and Pataki [5]: first, when the knapsack problem has an equality constraint, and so both reformulations were applicable, there was no difference in their performance. Second, Theorem 1, which asserts that branching on a single variable in the reformulation is equivalent to branching on px in the original problem is verified by another experiment: creating a new variable z, and adding the redundant constraint $z = px$ to the original formulations. Even without specifying a higher priority for branching on the z variable, the *original* instances with this addition solved as fast as the reformulations.

Acknowledgments

Mustafa Tural would like to thank Telcordia Technologies for their hospitality for the duration of this work.

REFERENCES

1. Lenstra HW Jr. Integer programming with a fixed number of variables. Math Oper Res 1983;8:538–548. (First announcement (1979).)
2. Kannan R. Improved algorithms for integer programming and related lattice problems. Proceedings of the 15th Annual ACM Symposium on Theory of Computing; Boston (MA). New York: The Association for Computing Machinery; 1983. pp. 193–206.
3. Kannan R. Minkowski's convex body theorem and integer programming. Math Oper Res 1987;12:415–440.
4. Aardal K, Hurkens CAJ, Lenstra AK. Solving a system of linear Diophantine equations with lower and upper bounds on the variables. Math Oper Res 2000;25:427–442.
5. Krishnamoorthy B, Pataki G. Column basis reduction and decomposable knapsack problems. Discrete Optim 2009;6:242–270.
6. Mehrotra S, Li Z. Branching on hyperplane methods for mixed integer linear and convex programming using adjoint lattices. J Glob Optim 2010. DOI: 10.1007/s10898-010-9554-4.
7. Jeroslow RG. Trivial integer programs unsolvable by branch-and-bound. Math Program 1974;6:105–109.
8. Aardal K, Lenstra AK. Hard equality constrained integer knapsacks. Math Oper Res 2004;29:724–738.
9. Aardal K, Lenstra AK. Erratum to: Hard equality constrained integer knapsacks. Math Oper Res 2006;31:846.
10. Pataki G, Tural M, Wong EB. Basis reduction and the complexity of branch-and-bound. Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). Austin (TX): SIAM; 2010. pp. 1254–1261
11. Babai L. On Lovász lattice reduction, and the nearest lattice point problem. Combinatorica 1986;6:1–13.
12. Land AH, Doig AG. An automatic method for solving discrete programming problems. Econometrica 1960;28:497–520.
13. Schrijver A. Theory of linear and integer programming. Chichester, UK: Wiley; 1986.
14. Lenstra AK, Lenstra HW Jr, Lovász L. Factoring polynomials with rational coefficients. Math Ann 1982;261:515–534.
15. LiDIA. A library for computational number theory. http://www.cdc.informatik.th-darmstadt.de/TI/LiDIA/.
16. Shoup V. NTL: A number theory library. http://www.shoup.net. 1990.
17. Korkine A, Zolotareff G. Sur les formes quadratiques. Math Ann 1873;6:366–389.
18. Lagarias JC, Lenstra HW, Schnorr CP. Korkine-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. Combinatorica 1990;10:333–348.
19. Ramirez Alfonsin JL. The Diophantine Frobenius problem. Oxford lecture series in mathematics and its applications. New York: Oxford University Press; 2005.
20. Cornuéjols G, Urbaniak R, Weismantel R, *et al*. Decomposition of integer programs and of generating sets. Volume 1284, Algorithms - ESA 1997. Lecture notes in computer science. London, UK: Springer; 1997. pp. 92–103.
21. Chvátal V. Hard knapsack problems. Oper Res 1980;28:1402–1411.
22. Furst M, Kannan R. Succinct certificates for almost all subset sum problems. SIAM J Comput 1989;18:550–558.
23. Lagarias JC, Odlyzko AM. Solving low-density subset sum problems. J ACM 1985;32:229–246.
24. Frieze A. On the Lagarias-Odlyzko algorithm for the subset sum problem. SIAM J Comput 1986;15:536–540.
25. Coster MJ, Joux A, LaMacchia BA, *et al*. Improved low-density subset sum algorithms. Comput Complex 1992;2:111–128.
26. Lovász L, Scarf HE. The generalized basis reduction algorithm. Math Oper Res 1992;17:751–764.
27. Kannan R. Lattice translates of a polytope and the Frobenius problem. Combinatorica 1992;12:161–177.
28. Barvinok AI. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. Math Oper Res 1994;19:769–779.
29. Dyer M, Kannan R. On Barvinok's algorithm for counting lattice points in fixed dimension. Math Oper Res 1997;22:545–549.
30. De Loera JA, Hemmecke R, Tauzer J, *et al*. Effective lattice point counting in rational convex polytopes. J Symb Comput 2004;38:1273–1302.
31. Koeppe M. A primal Barvinok algorithm based on irrational decompositions. SIAM J Discrete Math 2007;21:220–236.
32. Eisenbrand F. Fast integer programming in fixed dimension. 11th Annual European

Symposium on Algorithms - ESA 2003. Volume 2832, Lecture notes in computer science. Berlin, Germany: Springer; 2003. pp. 196–207.

33. Hoşten S, Sturmfels B. Computing the integer programming gap. Combinatorica 2007;27:367–382.

34. Eisenbrand F, Shmonin G. Parametric integer programming in fixed dimension. Math Oper Res 2008;33:839–850.

35. Kannan R. Algorithmic geometry of numbers. Ann Rev Comput Sci. Palo Alto (CA): 1987;2:231–267.

36. Aardal K, Eisenbrand F. Integer programming, lattices, and results in fixed dimension. Discrete optimization. Volume 12, Handbooks in operations research, and management science. Amsterdam, The Netherlands: Elsevier; 2005. pp. 171–243.

37. Eisenbrand F. Integer programming and algorithmic geometry of numbers. 50 Years of integer programming 1958–2008. Berlin, Germany: Springer; 2010. pp. 505–559.

38. Grötschel M, Lovász L, Schrijver A. Geometric algorithms and combinatorial optimization. Volume 2, Algorithms and combinatorics. 2nd corrected ed. Berlin, Germany: Springer; 1993.

39. Gao L, Zhang Y. Computational experience with Lenstra's algorithm. Technical Report #TR02–12. Department of Computational and Applied Mathematics, Rice University; 2002.

40. Cook W, Rutherford T, Scarf HE, *et al*. An implementation of the generalized basis reduction algorithm for integer programming. ORSA J Comput 1993;5:206–212.

41. Cornuéjols G, Dawande M. A class of hard small 0–1 programs. INFORMS J Comput 1999;11:205–210.

42. Aardal K, Bixby RE, Hurkens CAJ, *et al*. Market split and basis reduction: towards a solution of the Cornuéjols-Dawande instances. INFORMS J Comput 2000;12: 192–202.

43. Louveaux Q, Wolsey LA. Combining problem structure with basis reduction to solve a class of hard integer programs. Math Oper Res 2002;27:470–484.