Search for Combinatorial Objects Using Lattice Algorithms – Revisited

Alfred Wassermann(✉)

Department of Mathematics, University of Bayreuth, 95440 Bayreuth, Germany
alfred.wassermann@uni-bayreuth.de

Abstract. In 1986, Kreher and Radziszowski were the first who used the famous LLL algorithm to construct combinatorial designs. Subsequently, lattice algorithms have been applied to construct a large variety of objects in design theory, coding theory and finite geometry. Unfortunately, the use of lattice algorithms in combinatorial search is still not well established. Here, we provide a list of problems which could be tackled with this approach and give an overview on exhaustive search using lattice basis reduction. Finally, we describe a different enumeration strategy which might improve the power of this method even further.

Keywords: Lattice enumeration · Combinatorial search

1 Introduction

In 1982, Lenstra, Lenstra and Lovász published the groundbreaking LLL algorithm, which finds in polynomial time "short" vectors in a lattice. As soon as in 1983, Lagarias and Odlyzko [44] applied the LLL algorithm successfully to break certain cryptosystems based on the subset sum problem.

It seems that in the field of constructive combinatorics, Kreher and Radziszowski were the first who used the LLL algorithm. In [40,41] they used it to construct a 6-$(14, 7, 4)$ design. Subsequently, lattice basis reduction was used by the author and collaborators to find the first combinatorial designs for $t = 7$, 8, and 9 with small parameters and other variants of designs, see e.g. [2–4,14,42,43,45] (see [16] for a comprehensive overview on combinatorial design theory). The same program has been used successfully in the search for large sets of designs [5,46,47], in coding theory (linear codes, codes over rings, two-weight codes, covering codes) [6,12,32,34,37,54,62], subspace designs and their variants [8–11,13,33], as well as in finite geometry [7] and other problems.

All of these combinatorial search problems can be reduced to the solution of a Diophantine linear system which is a generalization of the subset sum problem studied by Lagarias and Odlyzko [44] and has the following form.

Let $A \in \mathbb{Z}^{m\times n}$, $\mathbf{d} \in \mathbb{Z}^m$, and $\mathbf{l}, \mathbf{r} \in \mathbb{Z}^n$. Determine all vectors $\mathbf{x} \in \mathbb{Z}^n$ such that

$$A \cdot \mathbf{x} = \mathbf{d} \ \text{ and } \ \mathbf{l} \le \mathbf{x} \le \mathbf{r}\,, \tag{1}$$

where $\mathbf{l} \le \mathbf{r}$ for vectors $\mathbf{l}, \mathbf{r} \in \mathbb{Z}^n$ is defined as $l_i \le r_i$ for all $0 \le i < n$.

With the substitution $\mathbf{x} := \mathbf{x} - \mathbf{l}$, $\mathbf{d} := \mathbf{d} - A \cdot \mathbf{l}$ and $\mathbf{r} := \mathbf{r} - \mathbf{l}$, it suffices to consider $\mathbf{l} = \mathbf{0}$ as a lower bound on the variables.

Kramer and Mesner [39] reduced the search for combinatorial designs with prescribed automorphism group to such a problem.

Solving equation (1) is a special instance of the *multi-dimensional subset sum problem* which is known to be NP-complete [22]. Since problem (1) can be reduced to many other NP-hard problems it is no surprise that there are many solving algorithms available. See [23,31,49] for a survey.

In case the right hand side vector $\mathbf{d}$ in (1) is the all-one vector, the problem is also called *exact cover problem* and the fastest algorithm seems to be the *dancing links* algorithm[1] by Knuth [35] or – in special cases – *maximum clique search*, see e.g. [53]. In case there is a "$\leq$" instead of "$=$" in (1), it seems that typical integer linear programming algorithms [24,27] are most promising.

However, in the special case that there is "$=$" in (1), and $\mathbf{d}$ is much larger than the all-one vector, and $\mathbf{r}$ is the all-one vector (i.e. solution vectors $\mathbf{x}$ are $\{0, 1\}$ vectors), reduction of the problem to a lattice point enumeration problem has been very successful as shown in the above references. The algorithm has been described in detail in [60,61] but unfortunately there are not many other implementations if any. This may be due to the widespread misconception that lattice basis reduction is only able to find random solutions which was the case in the implementation of [41]. It has been overlooked that lattice basis reduction can be followed by an exhaustive enumeration of all solutions of (1) with a backtracking algorithm.

In the sequel we will give an overview to exhaustive enumeration of all solutions of (1) using lattice point enumeration and also hint to a different enumeration scheme (limited discrepancy search) which allows to find the first solutions sometimes much more quickly.

2 Lattices

Let $\mathbb{R}^n$ denote the real Euclidean n-dimensional space. Its elements $\mathbf{v} \in \mathbb{R}^n$ are written as column vectors $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})^\top$. Let $\langle \mathbf{v}, \mathbf{w} \rangle = \sum_{i \in n} v_i \cdot w_i$ be the standard bilinear form for $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$. For $q \in \mathbb{R}$, $q \geq 1$, the ℓ_q-*norm* is defined by

$$\|{-}\|_q : \mathbb{R}^n \to \mathbb{R} : \mathbf{v} \mapsto \Big(\sum_{i \in n} |v_i|^q\Big)^{1/q},$$

and the ℓ_∞-*norm* is defined as:

$$\|{-}\|_\infty : \mathbb{R}^n \to \mathbb{R} : \mathbf{v} \mapsto \max_{i \in n} |v_i| \,.$$

Let $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ be m linearly independent vectors in $\mathbb{R}^n$. The *discrete additive subgroup* of $\mathbb{R}^n$

[1] Updated versions available at https://www-cs-faculty.stanford.edu/~knuth/programs.html.

$$\mathcal{L}(\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}) = \{ \sum_{i=0}^{m-1} u_i \cdot \mathbf{b}^{(i)} \mid u_i \in \mathbb{Z},\ i \in m \} \subset \mathbb{R}^n$$

is called *lattice* with *basis* $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$.

The *rank* m of a lattice $\mathcal{L}$ with basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ is the dimension of the $\mathbb{R}$-subspace $\langle \mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)} \rangle$ which is spanned by the basis. The corresponding $n \times m$-matrix

$$B = \left(\mathbf{b}^{(0)} \mid \ldots \mid \mathbf{b}^{(m-1)} \right)$$

is called a *generator matrix* of $\mathcal{L}$ if $\mathcal{L} = \mathcal{L}(\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)})$.

For a lattice $\mathcal{L} \subset \mathbb{R}^n$, the most important algorithmic problems are:

- *Shortest vector problem* (SVP): Find an ℓ_q-shortest vector in L, i.e. find an element $\mathbf{w}$ in $\mathcal{L}$ such that

$$\|\mathbf{w}\|_q = \min\{\|\mathbf{w}'\|_q \mid \mathbf{w}' \in \mathcal{L} \setminus \{0\}\}.$$

  This question is most interesting for the Euclidean norm ℓ_2, the ℓ_1-norm, and the ℓ_∞-norm.
- *Closest vector problem* (CVP): Given a vector $\mathbf{v} \in \mathbb{R}^n$, find a lattice vector $\mathbf{w} \in \mathcal{L}$ which is closest to $\mathbf{v}$ in the ℓ_q-norm, i.e. such that

$$\|\mathbf{v} - \mathbf{w}\|_q = \min\{\|\mathbf{v} - \mathbf{w}'\|_q \mid \mathbf{w}' \in \mathcal{L}\}\,.$$

- *Lattice basis reduction*: Given a basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ of the lattice $\mathcal{L}$ compute a new basis $\mathbf{b'}^{(0)}, \mathbf{b'}^{(1)}, \ldots, \mathbf{b'}^{(m-1)}$ of L consisting of "shortest" vectors. Here, the meaning of short will have to be made precise, compare Fig. 1.

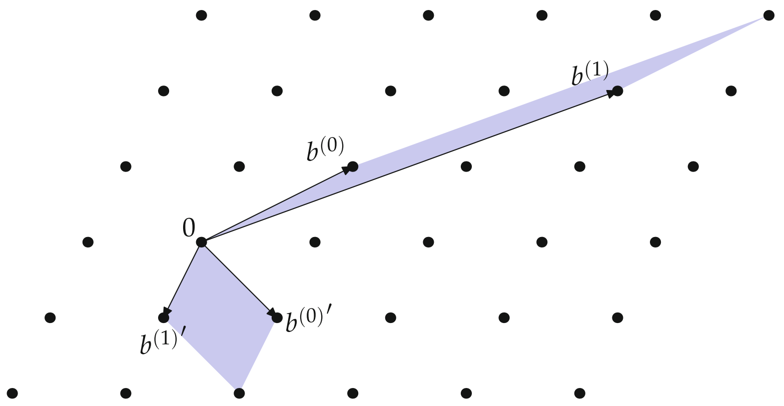

Fig. 1. Two different bases for $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}$ and $\mathbf{b}^{(0)'}, \mathbf{b}^{(1)'}$ of the same lattice

For an overview on the algorithmic complexity of the above problems we refer e.g. to [50] and the literature cited there.

Concerning the last of the mentioned problems, we remark that the problem of finding a basis consisting of shortest vectors is not exactly defined provided the dimension is at least three. In fact, many different versions of the concept of a shortest basis exist. Two classical concepts are the reduced bases in the sense of Minkowski [51] and the reduced quadratic forms in the sense of Korkine and Zolotarev [38] which rely on the computation of shortest lattice vectors in sublattices and related lattices. Therefore, the problem of computing a reduced lattice basis in the sense of Minkowski or Korkine and Zolotarev is at least as hard as the shortest vector problem.

Let $B = (\mathbf{b}^{(0)} \mid \ldots \mid \mathbf{b}^{(m-1)})$ be a generator matrix of a lattice $\mathcal{L}$. The matrix $G(B) = (\langle \mathbf{b}^{(i)}, \mathbf{b}^{(j)} \rangle)_{i,j \in m} \in \mathbb{R}^{m \times m}$ is called *Gram matrix* of the lattice basis. The volume of the lattice $\mathcal{L}$ is defined as $\mathrm{Vol}(\mathcal{L}) = \det(\mathcal{L}) = \sqrt{\det(G(B))}$, it does not depend on the choice of the basis. Further invariants of a lattice – independent from the choice of the basis – are the *successive minima* of Minkowski [51].

Let $\mathcal{L} \subset \mathbb{R}^n$ be a lattice of rank m. For an integer $i \in m$ let $\lambda_i(\mathcal{L})$ be the least positive real number for which there exist $i+1$ linearly independent lattice vectors $\mathbf{v} \in \mathcal{L} \setminus \{0\}$ with $\|\mathbf{v}\|_2 \leq \lambda_i(L)$. The numbers $\lambda_0(\mathcal{L}), \lambda_1(\mathcal{L}), \ldots, \lambda_{m-1}(\mathcal{L})$ are the *successive minima* of the lattice $\mathcal{L}$. From the definition it follows that $\lambda_0(\mathcal{L}) \leq \lambda_1(\mathcal{L}) \leq \ldots \leq \lambda_{m-1}(\mathcal{L})$. A classical result due to Hermite [26] gives an upper bound for the ℓ_2-shortest vector of a lattice $\mathcal{L} \subset \mathbb{Z}^n$, namely $\mathcal{L}$ contains a nonzero vector $\mathbf{v}$ such that

$$\|\mathbf{v}\|_2^2 \leq (4/3)^{(m-1)/2} \cdot \det(\mathcal{L})^{2/m} .$$

3 Lattice Basis Reduction

Let $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ be a set of linearly independent vectors $\in \mathbb{R}^n$.

Gram–Schmidt orthogonalization (GSO) is the orthogonal family defined for $0 \leq i < m$ by

$$\hat{\mathbf{b}}^{(i)} = \mathbf{b}^{(i)} - \sum_{j=0}^{i-1} \mu_{ij} \cdot \hat{\mathbf{b}}^{(j)} ,$$

where

$$\mu_{ij} = \frac{\langle \mathbf{b}^{(i)}, \hat{\mathbf{b}}^{(j)} \rangle}{\langle \hat{\mathbf{b}}^{(j)}, \hat{\mathbf{b}}^{(j)} \rangle} . \tag{2}$$

For $0 \leq t < m$ and $\mathbf{v} \in \mathbb{R}^n$ the *orthogonal projection* $\pi_t(\mathbf{v})$ is defined by

$$\pi_t : \mathbb{R}^n \to \langle \mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(t-1)} \rangle^{\perp}, \quad \mathbf{v} \mapsto \sum_{j=t}^{m-1} \frac{\langle \mathbf{v}, \hat{\mathbf{b}}^{(j)} \rangle}{\langle \hat{\mathbf{b}}^{(j)}, \hat{\mathbf{b}}^{(j)} \rangle} \cdot \hat{\mathbf{b}}^{(j)}$$

and $\hat{\mathbf{b}}^{(t)} = \pi_t(\mathbf{b}^{(t)})$. The orthogonal projection of a lattice $\mathcal{L}$ is the lattice $\mathcal{L}_t$ defined by

$$\mathcal{L}_t = \mathcal{L}(\pi_t(\mathbf{b}^{(t)}), \pi_t(\mathbf{b}^{(t+1)}), \ldots, \pi_t(\mathbf{b}^{(m-1)})) .$$

A basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ of a lattice $\mathcal{L} \subset \mathbb{R}^n$ is *reduced in the sense of Korkine and Zolotarev* [38] if

1. $\mathbf{b}^{(0)}$ is an ℓ_2-shortest vector in $\mathcal{L}$ and
2. for $0 \leq t < m$, $\hat{\mathbf{b}}^{(t)}$ is an ℓ_2-shortest vector in the lattice $\mathcal{L}_t(\mathbf{b}^{(t)}, \ldots, \mathbf{b}^{(m-1)})$.

However, no polynomial time algorithm to compute a Korkine–Zolotarev-reduced basis is known. A major breakthrough was achieved by Lenstra, Lenstra, and Lovász in their seminal work [48]. They compute a different type of reduced basis, which is now called an *LLL-reduced basis*, see the original paper [48] or textbooks like [15,52].

The *LLL algorithm* computes an LLL-reduced basis. The input is a basis $\mathbf{b}^{(0)}, \ldots, \mathbf{b}^{(m-1)}$ of the lattice $\mathcal{L}$ of rank m.

```
(1)  Let $\delta \in \mathbb{R}$ with $\frac{1}{4} < \delta < 1$.
(2)  Set $k := 0$.
(3)  do
(4)     1. for $j = 0, \ldots, k-1$
(5)          replace $\mathbf{b}^{(k)}$ by $\mathbf{b}^{(k)} - \lfloor \mu_{kj} \rceil \mathbf{b}^{(j)}$,
(6)            where $\mu_{kj}$ is the Gram-Schmidt coefficient (2).
(7)     2. if $\delta \|\pi_k(\mathbf{b}^{(k)})\|^2 > \|\pi_k(\mathbf{b}^{(k+1)})\|^2$ then
(8)          swap $\mathbf{b}^{(k+1)}$ and $\mathbf{b}^{(k)}$
(9)          update $\hat{\mathbf{b}}^{(k+1)}$, $\hat{\mathbf{b}}^{(k)}$ and $\mu$
(10)         set $k := \max(k-1, 0)$
(11)       else
(12)         set $k := k+1$
(13) until $k = m-1$.                                              □
```

The output $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ of the LLL-algorithm with $\frac{1}{4} < \delta < 1$ is called δ-reduced basis of the lattice $\mathcal{L}$.

The LLL algorithm runs in polynomial time in m, n, and the size of the entries of the basis vectors. In [52, Chapters 4 and 5] recent developments are described, e.g. how to approximate the LLL algorithm by using floating point numbers.

The LLL algorithms can not be expected to compute shortest vectors in a lattice. Let $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$ be a δ-reduced basis of the lattice $\mathcal{L} \subset \mathbb{Q}^n$. Then, the following bounds can be proved [48].

$$\|\mathbf{b}^{(j)}\|^2 \leq (\frac{4}{4\delta - 1})^i \cdot \|\hat{\mathbf{b}}^{(i)}\|^2 \ \text{ for } \ 0 \leq j \leq i < m\,. \tag{3}$$

$$\det(\mathcal{L}) \leq \prod_{i=0}^{m-1} \|\mathbf{b}^{(i)}\| \leq (\frac{4}{4\delta - 1})^{m(m-1)/4} \cdot \det(\mathcal{L})\,. \tag{4}$$

$$\|\mathbf{b}^{(0)}\| \leq (\frac{4}{4\delta - 1})^{(m-1)/4} \cdot \det(\mathcal{L})^{1/m}\,. \tag{5}$$

The fascinating mystery behind the LLL algorithm is that in many cases it produces a much better approximation to the shortest vector of the lattice than the proven bounds guarantee.

Nevertheless, a full reduction in the sense of Korkine and Zolotarev would require exponential complexity. In [56,57] *blockwise Korkine–Zolotarev reduction* (BKZ) was introduced which restricts enumeration in the sense of Korkine and Zolotarev to blocks of a fixed size β of basis vectors, i.e. searches by exhaustive enumeration for nontrivial integer linear combinations

$$u_k\mathbf{b}^{(k)} + u_{k+1}\mathbf{b}^{(k+1)} + \ldots + u_{k+\beta-1}\mathbf{b}^{(k+\beta-1)}$$

which minimize the Euclidean length of

$$\pi_k(u_k\mathbf{b}^{(k)} + u_{k+1}\mathbf{b}^{(k+1)} + \ldots + u_{k+\beta-1}\mathbf{b}^{(k+\beta-1)}) .$$

The original LLL algorithm can be interpreted as blockwise Korkine–Zolotarev reduction for $\beta = 2$. For a further description of improved practical versions and recent developments, e.g. sieving methods, we refer to [52,57,58]. In a blockwise Korkine–Zolotarev-reduced basis of a lattice of rank m the factor $(\frac{4}{4\delta-1})^{(m-1)/2}$ can be replaced by $(1+\epsilon)^m$ for any fixed $\epsilon > 0$. Of course, the time complexity increases exponentially as ϵ approaches 0.

4 Lattice Embedding of Diophantine Linear Systems

In [44], Lagarias and Odlyzko described the reduction of problem (1) for $\{0,1\}$ vectors $\mathbf{x}$, i.e. $\mathbf{r} = \mathbf{1}$. In [17,18] their embedding of (1) into a lattice problem was be improved. In turn, the following generalization to arbitrary upper bounds $\mathbf{r}$ has been given in [61].

The basis of the lattice to which the LLL algorithm is applied consists of the columns of the following generator matrix of size $(m+n+1)\times(n+1)$:

$$\left(\begin{array}{c|cccc}
-N\cdot d & & N\cdot A & & \\
\hline
-r_{\max} & 2c_1 & 0 & \cdots & 0 \\
-r_{\max} & 0 & 2c_2 & \cdots & 0 \\
\vdots & \vdots & & \ddots & \vdots \\
-r_{\max} & 0 & \cdots & \cdots & 2c_n \\
\hline
r_{\max} & 0 & \cdots & \cdots & 0
\end{array}\right) \tag{6}$$

where $N \in \mathbb{Z}_{>0}$ is a large constant and

$$r_{\max} = \operatorname{lcm}\{r_1, \ldots, r_n\} \quad \text{and} \quad c_i = \frac{r_{\max}}{r_i}\,, \quad 1 \le i \le n\,.$$

If N is large enough, see [1], the reduced basis will consist of $n-m+1$ vectors with only zeroes in the first m rows and m vectors which contain at least one nonzero entry in the first m rows. The latter vectors can be removed from the basis. From the remaining $n-m+1$ vectors we can delete the first m rows which contain only zeroes. This gives a basis $\mathbf{b}^{(0)}$, $\mathbf{b}^{(1)}$, …, $\mathbf{b}^{(n-m)} \in \mathbb{Z}^{n+1}$.

In the second step of the algorithm, see Sect. 5, all integer linear combinations of the basis vectors $\mathbf{b}^{(0)}$, $\mathbf{b}^{(1)}$, …, $\mathbf{b}^{(n-m)} \in \mathbb{Z}^{n+1}$ are enumerated which correspond to solutions of (1).

Theorem 1 ([61]). *Let*

$$\mathbf{w} = u_0 \cdot \mathbf{b}^{(0)} + u_1 \cdot \mathbf{b}^{(1)} + \ldots + u_{n-m} \cdot \mathbf{b}^{(n-m)} \tag{7}$$

be an integer linear combination of the basis vectors with $w_0 = r_{\max}$.
$\mathbf{w}$ *is a solution of the system* (1) *if and only if*

$$\mathbf{w} \in \mathbb{Z}^{n+1} \text{ where } -r_{\max} \le w_i \le r_{\max},\ 1 \le i \le n\,.$$

5 Lattice Point Enumeration

Usually, we are interested in finding all solutions to problem (1), or to conclude that there are none. In terms of the associated lattice (6), this mean that we wish to enumerate all lattice points which are subject to a certain set of constraints. Such an approach has first been described by Ritter [55] for $\{0,1\}$ problems. Here we solve the general problem with arbitrary bounds on the variables.

A priori, a lattice $\mathcal{L} = \{\sum_{i\in m} u_i \mathbf{b}^{(i)} \mid u_i \in \mathbb{Z}\}$ of rank m contains infinitely many elements. However, it will turn out that there are bounds on the integers $|u_i|$, $i \in m$, which depend solely on the lattice basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m-1)}$. These bounds reduce the problem of finding solution vectors to a finite set of lattice vectors. Each solution vector $\mathbf{v}$ has the upper bounds

$$\|\mathbf{v}\|_2^2 \le (n+1) \cdot r_{\max}^2 \quad \text{and} \quad \|\mathbf{v}\|_\infty \le r_{\max}\,.$$

The exhaustive enumeration is arranged as backtracking algorithm. Starting from $u_{n-m} \in \mathbb{Z}$, successively all possible $u_t \in \mathbb{Z}$ for $t = n-m, n-m-1, \ldots, 1, 0$ are tested. The enumeration can be pruned at stage t if certain conditions are violated. These pruning tests have quite a long history and are based on the work of [19–21,28,29,36,55].

In each level t of the backtracking algorithm, $\mathbf{w}^{(t)} = \pi_t(\sum_{j=t}^{n-m} u_j \mathbf{b}^{(j)})$ is the projection of the linear combination of the already fixed variables u_t, u_{t+1}, …, u_{n-m} into the subspace of $\mathbb{R}^{n+1}$ which is orthogonal to the linear span $\langle b_0, \ldots, b_{t-1}\rangle$.

Starting from $\mathbf{w}^{(n-m+1)} = \mathbf{0}$, $\mathbf{w}^{(t)}$ can be computed iteratively from $\mathbf{w}^{(t+1)}$ by

$$\mathbf{w}^{(t)} = (\sum_{i=t}^{n-m} u_i \mu_{it}) \hat{\mathbf{b}}^{(t)} + \mathbf{w}^{(t+1)}$$

with Gram-Schmidt coefficients μ_{it}. In each level t, $n-m \ge t \ge 0$, all possible integer values for the variable u_t are tested. The following two main tests allow to restrict the possible values of u_t.

First pruning condition. For all $j \le t$ the vectors $\hat{\mathbf{b}}^{(j)}$ are orthogonal to $\mathbf{w}^{(t+1)}$ and therefore

$$\|\mathbf{w}^{(t)}\|_2^2 = (\sum_{i=t}^{n-m} u_i \mu_{it})^2 \|\hat{\mathbf{b}}^{(t)}\|_2^2 + \|\mathbf{w}^{(t+1)}\|_2^2\,.$$

We notice that $\mathbf{w}^{(0)} = \sum_{j=0}^{n-m} u_j \mathbf{b}^{(j)}$. Using $\|\mathbf{w}^{(j)}\|_2 \geq \|\mathbf{w}^{(t)}\|_2$ for $j \leq t$ we can backtrack as soon as

$$\|\mathbf{w}^{(t)}\|_2^2 > c := (n+1) \cdot r_{\max}^2 \,.$$

For fixed $u_{t+1}, \ldots, u_{n-m}$, this gives a bound for u_t:

$$(u_t + \sum_{i=t+1}^{n-m} u_i \mu_{it})^2 \leq \frac{c - \|\mathbf{w}^{(t+1)}\|_2^2}{\|\hat{\mathbf{b}}^{(t)}\|_2^2} \,.$$

Second pruning condition. The second test is an adaption to the special situation that we are searching for an integer linear combination of the basis vectors which consists solely of components whose absolute value is bounded by $r_{\max}$. It is based on the following theorem by Ritter [55].

Theorem 2 ([55]). *If the given sequence of integers $u_t, u_{t+1}, \ldots, u_{n-m} \in \mathbb{Z}$ can be extended to $u_0, \ldots, u_t, \ldots, u_{n-m} \in \mathbb{Z}$ such that $\sum_{i=0}^{n-m} u_i \mathbf{b}^{(i)}$ is a solution of (1), then for all $y_t, y_{t+1}, \ldots, y_{n-m} \in \mathbb{R}$:*

$$|\sum_{i=t}^{n-m} y_i \|\mathbf{w}^{(i)}\|_2^2| \leq r_{\max} \cdot \| \sum_{i=t}^{n-m} y_i \mathbf{w}^{(i)} \|_1 \,.$$

We use this theorem in the enumeration algorithm in the following way. Taking $(y_t, y_{t+1}, \ldots, y_{n-m}) = (1, 0, \ldots, 0)$ results in the test

$$\|\mathbf{w}^{(t)}\|_2^2 \leq r_{\max} \|\mathbf{w}^{(t)}\|_1 \,.$$

If this inequality is violated for some vector $\mathbf{w}^{(t)} = x\hat{\mathbf{b}}^{(t)} + \mathbf{w}^{(t+1)}$, then it will also fail for all vectors of the form $(x+r)\hat{\mathbf{b}}^{(t)} + \mathbf{w}^{(t+1)}$ with $r \in \mathbb{Z}$ and $xr > 0$.

Summarizing, a high level description of the algorithm to solve (1) is as follows.

Lattice point enumeration

Given the generator matrix (6) of the lattice $\mathcal{L} \subset \mathbb{R}^{m+n+1}$ of rank $n+1$ of problem (1) all nonzero vectors $\mathbf{v} \in \mathcal{L}$ such that $\|\mathbf{v}\|_\infty \leq r_{\max}$ are determined.

- Compute an LLL/BKZ-reduced basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(n)}$ of the lattice $\mathcal{L}$.
- Delete the unnecessary columns and rows of the generator matrix. The remaining basis $\mathbf{b}^{(0)}, \mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(n-m)} \subset \mathbb{R}^{n+1}$ has rank $n-m+1$.
- Compute the Gram–Schmidt vectors $\hat{\mathbf{b}}^{(0)}, \hat{\mathbf{b}}^{(1)}, \ldots, \hat{\mathbf{b}}^{(n-m)}$ together with the Gram–Schmidt coefficients μ_{ij}.
- Set $R := (n+1) \cdot r_{\max}^2$.
- The recursive backtracking algorithm enum() is initiated with the call of enum$(n-m, \mathbf{0})$.

```
(1)   function enum($t$, $\mathbf{w}'$)
(2)   begin
(3)     onedirection := false
```

```
(4)        $y_t := \sum_{i=t+1}^{n-m} u_i \mu_{it}$
(5)        $u_t := \lfloor -y_t \rceil$
(6)        while true
(7)           $\mathbf{w} := (\sum_{i=t}^{n-m} u_i \mu_{it}) \hat{\mathbf{b}}^{(t)} + \mathbf{w}'$
(8)           if $\|\mathbf{w}\|_2^2 > R$ then return             /* step back */
(9)           if $t > 0$ then
(10)             if $\|\mathbf{w}\|_2^2 > r_{\max} \cdot \|\mathbf{w}\|_1$ then
(11)                if onedirection then return      /* step back */
(12)                else
(13)                   next($u_t$)
(14)                   onedirection := true
(15)                   goto line (7)
(16)                end if
(17)             else
(18)                enum($t-1$, $\mathbf{w}$)                    /* step forward */
(19)          else                                        /* $t = 0 \rightarrow$ solution */
(20)             output solution $w$
(21)          next($u_t$)
(22)       end while
(23)    end
```

The procedure next() in lines (13) and (21) determines the next possible integer value of the variable u_t. Initially, when entering a new level t, in line (5) u_t is set to be the closest integer value of $-y_t := -\sum_{i=t+1}^{n-m} u_i \mu_{it}$, say u_t^1. The next value u_t^2 of u_t is the second closest integer to $-y_t$ followed by u_t^3 and so forth. In other words, the values of u_t alternate with increasing distance around $-y_t$.

If the condition in line (10) is true then we do one more regular call of the procedure next() in line (13), i.e. u_t is set to be the next closest integer to $-y_t$. In Fig. 2 this happens while u_t^4 is determined. After that, the enumeration proceeds only in this remaining direction, see the computation of u_t^5 in Fig. 2. Finally, the second time when the condition in line (10) is true, the algorithm steps back and increases the enumeration level, see line (11).

6 Limited Discrepancy Search

For some problems of the form (1) it might be not interesting or may be impossible to enumerate all solutions. But nevertheless one is interested to find at least one solution as quick as possible. It turns out that in this situation, the enumeration algorithm in the previous section might not be optimal. In the following we will try to motivate a different enumeration algorithm.

The enumeration algorithm in Sect. 5 performs depth first search. In particular, when entering enumeration level t, u_t is chosen for $\mathbf{w} := (\sum_{i=t}^{n-m} u_i \mu_{it}) \hat{\mathbf{b}}^{(t)} + \mathbf{w}'$ in line (7) such that $\|\mathbf{w}\|_2$ is minimal among all choices for u_t. In other words, the depth first search is organized using the heuristic that choosing in each level

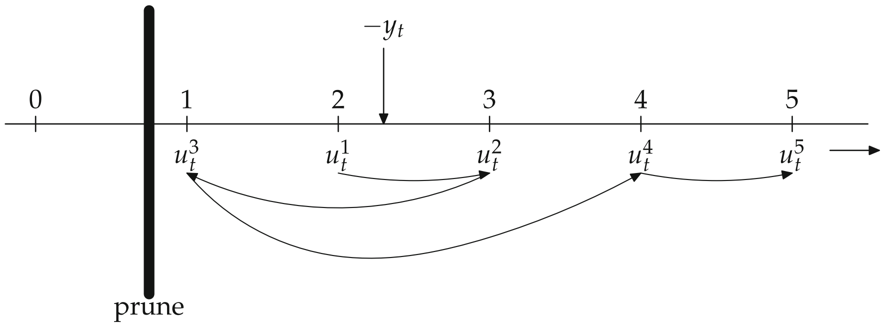

Fig. 2. Enumeration in level t and pruning after u_t^3

the vector $\mathbf{w}$ such that $\|\mathbf{w}\|_2$ is minimal will most probably lead to a solution vector. However, it may be that this choice for u_t in one of the first levels might lead to no solution, but nevertheless the algorithm will enumerate a huge search tree below u_t.

This is a general problem of depth first search algorithm. In 1995, Harvey and Ginsberg [25] described a simple, novel enumeration scheme called *limited discrepancy search* which aims to overcome this weakness of depth first search.

Assume that a backtrack algorithm has to examine a search tree. Each level corresponds to a variable and the algorithm has to assign a value to that variable, followed by a test if this assignment might lead to a solution. If yes, we can proceed to the next level, otherwise we have to assign a different value. If we have tested all values, we have to step back to the previous level. If values could be assigned to all variables, a solution has been found.

We assume that variable ordering is fixed and in each level of the backtracking there exists a heuristic which determines the order in which the values are assigned to the variable corresponding to that enumeration level. A *discrepancy* is defined as an deviation from the heuristic.

Harvey and Ginsberg suggest to enumerate the search tree in increasing number of discrepancies. In the first step, only the optimal choice in each level of enumeration in Sect. 5 is assigned to the variables until there is a contradiction or a solution is found. In the next step, all possible paths in the search tree with exactly one deviation (i.e. discrepancy) from the heuristic are examined. After that, all paths in the search tree with two deviations from the optimal choice are enumerated, and so forth.

In [25], the algorithm is given for binary search trees. In [30], the algorithm is described for general search trees, also a stop condition is given which allows to use the algorithm for exhaustive enumeration. The latter is mostly interesting to show the non-existence of solutions. There are other variants, see e.g. [59] for an overview.

Limited discrepancy search requires higher book keeping efforts than depth first search. Therefore, enumerating the whole search tree with limited discrepancy search will always be slower than with depth first search. But first tests with the lattice point enumeration algorithm and its value order heuristic in Sect. 5 show sometimes dramatic improvements for finding the first solution in hard combinatorial search problems mentioned in the introduction. A more detailed comparison of the two enumeration algorithms is in preparation.

It may be remarked that limited discrepancy search can also be useful for the enumeration algorithm in blockwise Korkine–Zolotarev reduction.

References

1. Aardal, K., Hurkens, C., Lenstra, A.K.: Solving a linear diophantine equation with lower and upper bounds on the variables. In: Bixby, R.E., Boyd, E.A., Ríos-Mercado, R.Z. (eds.) IPCO 1998. LNCS, vol. 1412, pp. 229–242. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-69346-7_18
2. Betten, A., Kerber, A., Laue, R., Wassermann, A.: Simple 8-designs with small parameters. Des. Codes Crypt. **15**, 5–27 (1998)
3. Betten, A., Kerber, A., Kohnert, A., Laue, R., Wassermann, A.: The discovery of simple 7-designs with automorphism group $P\Gamma L(2, 32)$. In: Cohen, G., Giusti, M., Mora, T. (eds.) AAECC 1995. LNCS, vol. 948, pp. 131–145. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-60114-7_10
4. Betten, A., Klin, M., Laue, R., Wassermann, A.: Graphical t-designs. Discrete Math. **197**(198), 111–121 (1999)
5. Betten, A., Laue, R., Wassermann, A.: New t-designs and large sets of t-designs. Discrete Math. **197**(198), 83–109 (1999)
6. Bouyukliev, I., Bouyuklieva, S., Kurz, S.: Computer classification of linear codes. CoRR abs/2002.07826 (2020). https://arxiv.org/abs/2002.07826
7. Braun, M., Kohnert, A., Wassermann, A.: Construction of (n, r)-arcs in PG$(2, q)$. Innovations Incidence Geom. **1**, 133–141 (2005)
8. Braun, M., Kerber, A., Laue, R.: Systematic construction of q-analogs of t-(v, k, λ)-designs. Des. Codes Crypt. **34**(1), 55–70 (2005). https://doi.org/10.1007/s10623-003-4194-z
9. Braun, M., Kiermaier, M., Kohnert, A., Laue, R.: Large sets of subspace designs. J. Comb. Theory Ser. A **147**, 155–185 (2017). https://doi.org/10.1016/j.jcta.2016.11.004
10. Braun, M., Kiermaier, M., Wassermann, A.: Computational methods in subspace designs. In: Greferath, M., Pavčević, M.O., Silberstein, N., Vázquez-Castro, M.Á. (eds.) Network Coding and Subspace Designs. SCT, pp. 213–244. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-70293-3_9
11. Braun, M., Kohnert, A., Östergård, P.R.J., Wassermann, A.: Large sets of t-designs over finite fields. J. Comb. Theory A **124**, 195–202 (2014)
12. Braun, M., Kohnert, A., Wassermann, A.: Optimal linear codes from matrix groups. IEEE Trans. Inform. Theory **51**(12), 4247–4251 (2005). https://doi.org/10.1109/TIT.2005.859291
13. Buratti, M., Kiermaier, M., Kurz, S., Nakić, A., Wassermann, A.: q-analogs of group divisible designs. In: Pseudorandomness and Finite Fields, Radon Series on Computational and Applied Mathematics, vol. 23. DeGruyter (2019)
14. Buratti, M., Wassermann, A.: On decomposability of cyclic triple systems. Australas. J. Comb. **71**(2), 184–195 (2018)
15. Cohen, H.: A Course in Computational Algebraic Number Theory. Graduate Texts in Mathematics, vol. 138. Springer, Berlin (1993). https://doi.org/10.1007/978-3-662-02945-9

16. Colbourn, C.J., Dinitz, J.H.: Handbook of Combinatorial Designs, Second Edition (Discrete Mathematics and Its Applications). Chapman & Hall/CRC, Boca Raton (2006)
17. Coster, M., Joux, A., LaMacchia, B., Odlyzko, A., Schnorr, C., Stern, J.: Improved low-density subset sum algorithms. Comput. Complex. **2**, 111–128 (1992)
18. Coster, M.J., LaMacchia, B.A., Odlyzko, A.M., Schnorr, C.P.: An improved low-density subset sum algorithm. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 54–67. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-46416-6_4
19. Coveyou, R., MacPherson, R.: Fourier analysis of uniform random number generators. J. ACM **14**, 100–119 (1967)
20. Dieter, U.: How to calculate shortest vectors in a lattice. Math. Comput. **29**(131), 827–833 (1975)
21. Fincke, U., Pohst, M.: Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. Math. Comput. **44**, 463–471 (1985)
22. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company, New York (1979)
23. Gibbons, P.B., Östergård, P.R.J.: Computational methods in design theory. In: Colbourn, C.J., Dinitz, J.H. (eds.) Handbook of Combinatorial Designs, chap. VII.6, 2 edn, pp. 755–783. Chapman & Hall/CRC, Boca Raton (2007)
24. Gurobi Optimization: Gurobi optimizer reference manual (2016). http://www.gurobi.com
25. Harvey, W.D., Ginsberg, M.L.: Limited discrepancy search. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI 1995, vol. 1. pp. 607–613. Morgan Kaufmann Publishers Inc., San Francisco (1995)
26. Hermite, C.: Extraits de lettres de M.Ch. Hermite à M. Jacobi sur différents objets de la théorie des nombres. J. reine angew. Math. **40**, 279–290 (1850)
27. IBM: ILOG CPLEX Optimizer (2010)
28. Kaib, M., Ritter, H.: Block reduction for arbitrary norms. Preprint, Universität Frankfurt (1995)
29. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**, 415–440 (1987)
30. Karoui, W., Huguet, M.-J., Lopez, P., Naanaa, W.: YIELDS: a yet improved limited discrepancy search for CSPs. In: Van Hentenryck, P., Wolsey, L. (eds.) CPAIOR 2007. LNCS, vol. 4510, pp. 99–111. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72397-4_8
31. Kaski, P., Östergård, P.R.: Classification Algorithms for Codes and Designs. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-28991-7
32. Kiermaier, M., Kurz, S., Solé, P., Stoll, M., Wassermann, A.: On strongly walk regular graphs, triple sum sets and their codes. ArXiv e-prints, **abs/1502.02711** (2020)
33. Kiermaier, M., Laue, R., Wassermann, A.: A new series of large sets of subspace designs over the binary field. Des. Codes Crypt. **86**(2), 251–268 (2018). https://doi.org/10.1007/s10623-017-0349-1
34. Kiermaier, M., Wassermann, A., Zwanzger, J.: New upper bounds on binary linear codes and a $\mathbb{Z}_4$-code with a better-than-linear Gray image. IEEE Trans. Inf. Theory **62**(12), 6768–6771 (2016). https://doi.org/10.1109/TIT.2016.2612654
35. Knuth, D.E.: Dancing links. In: Davies, J., Roscoe, B., Woodcock, J. (eds.) Millennial Perspectives in Computer Science: Proceedings of the 1999 Oxford-Microsoft Symposium in Honour of Sir Tony Hoare. Palgrave (2000)

36. Knuth, D.: The Art of Computer Programming, Vol. 2: Seminumerical Algorithms. Addison-Wesley, Reading (1969)
37. Kohnert, A.: Constructing two-weight codes with prescribed groups of automorphisms. Discret. Appl. Math. **155**(11), 1451–1457 (2007). https://doi.org/10.1016/j.dam.2007.03.006
38. Korkine, A., Zolotareff, G.: Sur les formes quadratiques. Math. Ann. **6**, 366–389 (1873)
39. Kramer, E.S., Mesner, D.M.: t-designs on hypergraphs. Discret. Math. **15**(3), 263–296 (1976). https://doi.org/10.1016/0012-365X(76)90030-3
40. Kreher, D.L., Radziszowski, S.P.: The existence of simple 6-$(14, 7, 4)$ designs. J. Comb. Theory Ser. A **43**, 237–243 (1986)
41. Kreher, D.L., Radziszowski, S.P.: Finding simple t-designs by using basis reduction. Congr. Numer. **55**, 235–244 (1986)
42. Krčadinac, V.: Some new designs with prescribed automorphism groups. J. Comb. Des. **26**(4), 193–200 (2018). https://doi.org/10.1002/jcd.21587
43. Krčadinac, V., Pavčević, M.O.: New small 4-designs with nonabelian automorphism groups. In: Blömer, J., Kotsireas, I.S., Kutsia, T., Simos, D.E. (eds.) MACIS 2017. LNCS, vol. 10693, pp. 289–294. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-72453-9_23
44. Lagarias, J., Odlyzko, A.: Solving low-density subset sum problems. J. Assoc. Comp. Mach. **32**, 229–246 (1985). Appeared already in Proceedings of 24th IEEE Symposium on Foundations of Computer Science, pp. 1–10 (1983)
45. Laue, R.: Constructing objects up to isomorphism, simple 9-designs with small parameters. In: Betten, A., Kohnert, A., Laue, R., Wassermann, A. (eds.) Algebraic Combinatorics and Applications, pp. 232–260. Springer, Heidelberg (2001). https://doi.org/10.1007/978-3-642-59448-9_16
46. Laue, R., Magliveras, S., Wassermann, A.: New large sets of t-designs. J. Comb. Des. **9**, 40–59 (2001)
47. Laue, R., Omidi, G.R., Tayfeh-Rezaie, B., Wassermann, A.: New large sets of t-designs with prescribed groups of automorphisms. J. Combin. Des. **15**(3), 210–220 (2007). https://doi.org/10.1002/jcd.20128
48. Lenstra, A., Lenstra Jr., H., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. **261**, 515–534 (1982)
49. Mathon, R.: Computational methods in design theory. In: Keedwell, A.D. (ed.) Surveys in Combinatorics, Proc. 13th Br. Comb. Conf., Guildford/UK 1991, vol. 166, pp. 101–117. London Mathematical Society Lecture Note (1991)
50. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems. Kluwer Academic Publishers (2002)
51. Minkowski, H.: Geometrie der Zahlen. Teubner, Leipzig (1896)
52. Nguyen, P.Q., Vallée, B.: The LLL Algorithm: Survey and Applications, 1st edn. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02295-1
53. Niskanen, S., Östergård, P.R.J.: Cliquer user's guide, version 1.0. Technical report T48, Helsinki University of Technology (2003)
54. Östergård, P.R.J., Quistorff, J., Wassermann, A.: New results on codes with covering radius 1 and minimum distance 2. Des. Codes Crypt. **35**, 241–250 (2005)
55. Ritter, H.: Aufzählung von kurzen Gittervektoren in allgemeiner Norm. Ph.D. thesis, Universität Frankfurt (1997)
56. Schnorr, C.: A hierachy of polynomial time lattice basis reduction algorithms. Theoret. Comput. Sci. **53**, 201–224 (1987)

57. Schnorr, C.P., Euchner, M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. In: Budach, L. (ed.) FCT 1991. LNCS, vol. 529, pp. 68–85. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-54458-5_51
58. Schnorr, C.P., Hörner, H.H.: Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 1–12. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-49264-X_1
59. van Beek, P.: Backtracking search algorithms. In: Rossi, F., van Beek, P., Walsh, T. (eds.) Handbook of Constraint Programming, Foundations of Artificial Intelligence, vol. 2, pp. 85–134. Elsevier (2006). https://doi.org/10.1016/S1574-6526(06)80008-8
60. Wassermann, A.: Finding simple *t*-designs with enumeration techniques. J. Comb. Des. **6**(2), 79–90 (1998)
61. Wassermann, A.: Attacking the market split problem with lattice point enumeration. J. Comb. Optim. **6**, 5–16 (2002)
62. Wassermann, A.: Computing the minimum distance of linear codes. In: Eighth International Workshop Algebraic and Combinatorial Coding Theory (ACCT VIII), Tsarskoe Selo, Russia, pp. 254–257 (2002)