Journal of Applied Mathematics, Statistics, and Computational Sciences
JAMSCS ISSN

OA
OPEN ACCESS

Copyright © Authors, 2026

Research Article

Techniques in Lattice Basis Reduction

Bal K. Khadka[1] and Spyros S. Magliveras[2]

Received: 2026-07-14
Revised: 2026-09-21
Accepted: 2026-10-01
Published: 2026-10-21

Article: 001
DOI:

Open access under the Creative Commons Attribution (CC BY 4.0) license.

[1]Department of Mathematics, Georgia Military College, Martinez, GA 30907, USA; bkhadka@gmc.edu
[2]Department of Mathematical Sciences, Florida Atlantic University, Boca Raton, FL 33431; spyros@fau.edu

Abstract

The shortest vector (SVP) and closest vector (CVP) problems, presently considered intractable, are algorithmic tasks that lie at the core of many theoretical and applied problems. We consider the LLL algorithm as a *black box* lattice basis reduction component, and the degree to which the shortest vector of the output basis compares with the Minkowski or weaker bounds as a measure of the *potency* of a basis reduction step. We make the assumption that a single LLL operation costs little, and that the payback of a very short vector is high, particularly if the vector satisfies the Minkowski bound. In this paper, we explore the effect of permuting the input basis B on the potency of a subsequent LLL reduction, and establish experimentally the sensitivity of LLL to permutations of B. In particular, we observe that permutations of large Hamming distance from the identity are more efficient in increasing the reduction potency. Regarding these factors, we investigate the efficacy of new optimization techniques to be used along with the LLL algorithm. The techniques we propose are: (i) *hill climbing (HC)* and (ii) *lattice diffusion-sublattice fusion (LDSF)*. The first technique relies on the sensitivity of LLL to permutations of the input basis B, and optimization ideas over the symmetric group Σ_m viewed as a metric space. The second technique relies on partitioning the lattice into sublattices, performing basis reduction on the sublattice blocks, fusing the sublattices, and repeating. Our approach provides better results for finding shortest vectors than LLL and other traditional methods.

Keywords: Lattice bases; unimodular matrices; right permutations; LLL algorithm; BKZ algorithm

1 Introduction: Historical Background

The development of lattice basis reduction began in 1773 by L. Lagrange [24], C. F. Gauss [9], C. Hermite [17] (1850), A. Korkin and Y. R. Zolotarev [21] (1873), H. Minkowski [26] (1896), and B. L. van der Waerden [35] (1956). The Hermite, Korkin, Zolotarev (HKZ [14]) lattice basis reduction method is assumed to be the strongest but impractical reduction approach. In 1982, Arjen Lenstra, Hendrik Lenstra and László Lovász [25], introduced a polynomial time basis reduction algorithm, presently known as LLL, to factor a rational polynomial. LLL approximates the *shortest vector* within a factor of $2^{n/2}$, where n is the rank of the lattice. Thus, LLL is fast but weak due to the large approximation factor. In 1986, L. Babai [3], proposed an algorithm that approximates the *closest vector* to a lattice point by a factor of $\left(\frac{3}{\sqrt{2}}\right)^n$. In 1987, C. P. Schnorr [30] proposed a further refinement of the LLL algorithm that improved the shortest vector approximation factor to $(1+\varepsilon)^n$. The *Block Korkin Zolotarev* (BKZ) algorithm was introduced and studied [3, 31] by Babai and Schnorr respectively. In 1993, M. Seysen [28] showed that each full rank n lattice $\mathscr{L}$ has a basis B and an associated basis B^* of the reciprocal lattice $\mathscr{L}^*$, with $S(B) \leq \exp(c_2 \cdot (\ln n)^2)$, where $S(B) = \max_{1\leq i\leq n}(\|b_i\| \cdot \|b_i^*\|)$ and c_2 independent of n. This was an improvement on the relation $S(B) \leq \exp(c_1 \cdot n^{1/3})$ established by J. Hastard and J. C. Lagarias [16]. In 1994, Schnorr and Euchner [32] introduced improved algorithms for BKZ and *deep insertion*, but the runtime of these algorithms is not polynomial. In 2001, M. Ajtai et al., [2] introduced a *Sieve* algorithm for the shortest lattice vector problem. In *Eurocrypt* 2008, *N. Gama and P. Nguyen* [13] observed experimentally that BKZ's practical runtime seems to grow exponentially with lattice dimension. In 2010, O. Dagdelen and M. Schneider, [8] presented a parallel version of the *lattice enumeration algorithm* using a multi-core CPU system. The shortest vector (SVP) and closest vector (CVP) problems, presently considered intractable, are algorithmic tasks that lie at the core of many number theoretic problems, integer programming, finding irreducible factors of polynomials, minimal polynomials of algebraic numbers, and simultaneous Diophantine approximation.

Lattice basis reduction has deep and extensive connections with modern cryptography, and cryptanalysis particularly in the

post-quantum era. At present, lattice based cryptosystems do not appear to be vulnerable to quantum attacks. Also, an implementation of a lattice based cryptosystem is faster than RSA or an ECC based system, at an equivalent security level, as proposed by T. Güneysu et al. in 2012, [15] and by J. Howe in 2015, [19].

In 1982, A. Shamir [33] proposed a polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem. In 1985, J. C. Lagarias and A. M. Odlyzko, [22] solved low density knapsack problems. In 1996, Hoffstein et al., [18] proposed NTRU, a ring theory based public key cryptosystem. In 1997, D. Coppersmith [6] published a paper on small solutions to polynomial equations, exploiting low exponent RSA vulnerabilities. In this paper, Coppersmith showed how to find small integer solutions to a modular polynomial equation in a single variable and extended the method to the multivariate case. In 1997, M. Ajtai and C. Dwork [1] designed a probabilistic public key cryptosystem whose security relies on the hardness of lattice problems. Their cryptosystem is quite impractical because of the massive data expansion, as it encrypts data bit-by-bit. Inspired by the Ajtai-Dwork cryptosystem, O. Goldreich et al., [12] proposed a public key cryptosystem based on the closest vector problem in a lattice, and J.-Y. Cai and T. W. Cusick, [5] introduced similar work on a lattice based public key cryptosystem in 1999. In 2001, C. Gentry, [10] published a paper on Key Recovery and Message Attacks on NTRU-Composite, where he recovers information about the private key used in NTRU. In 2001, D. Coppersmith and A. Shamir, [7] proposed a new lattice basis reduction technique to cryptanalyze the NTRU scheme. In 2009, Craig Gentry, [11], in his Ph.D. thesis, described a fully homomorphic encryption scheme based on ideal lattices. Currently, there are many mathematicians vigorously contributing to lattice basis reduction theory, and improving best lattice bounds.

2 Preliminaries

Let $B = \{b_1, b_2, ..., b_m\}$ be a set of m linearly independent row vectors in $\mathbb{R}^n$, where $m \leq n$. The *lattice* $\mathscr{L} \subset \mathbb{R}^n$, generated by B, is the set of all integral linear combinations of the vectors of B, that is:

$$\mathscr{L} = \{a_1 b_1 + a_2 b_2 + \cdots + a_m b_m : a_i \in \mathbb{Z}\}.$$

B is called a *basis* for $\mathscr{L}$, and m the *rank* or *dimension* of $\mathscr{L}$. If $m = n$, then $\mathscr{L}$ is called a *full-rank* lattice. Therefore, an arbitrary lattice vector $v = (v_1, \ldots, v_n) \in \mathscr{L}$ is given by $(v_1, \ldots, v_n) = (a_1, \ldots, a_m)B$, where $a_i \in \mathbb{Z}$. If $B = \{b_1, b_2, ..., b_m\}$ is a set of m linearly independent vectors on $\mathbb{R}^n$, as above, we denote the *span* of B over $\mathbb{R}$ by:

$$\langle B \rangle_{\mathbb{R}} = \Big\{ \sum_{i=1}^{m} x_i b_i : \ x_i \in \mathbb{R} \Big\}.$$

Similarly, we denote the span of B over $\mathbb{Z}$ by:

$$\langle B \rangle_{\mathbb{Z}} = \Big\{ \sum_{i=1}^{m} x_i b_i : \ x_i \in \mathbb{Z} \Big\} = \mathscr{L} \ .$$

The *weight of lattice basis* $B = \{b_1, b_2, \ldots, b_m\}$ is defined as the product of the lengths of all the basis vectors. That is, $\mathrm{wt}(B) = \prod_{i=1}^{m} \|b_i\|$.

The *determinant of the lattice* $\mathscr{L}$ is defined by

$$\det(\mathscr{L}) = \sqrt{\det(BB^t)}.$$

Lattices are invariant under unimodular transformations. That is, for any two bases B and C of the lattice $\mathscr{L}$ of rank m, there is an $m \times m$ unimodular matrix U such that $B = UC$. A square integer matrix U is said to be unimodular if $\det(U) = \pm 1$. There exist infinitely many unimodular matrices of size $m \times m$ for $m \geq 2$. Therefore, a lattice of rank $m \geq 2$ is generated by infinitely many lattice bases.

2.1 Geometry of numbers

Let $\mathscr{L}$ be a rank-m lattice in n-dimensional Euclidean space $\mathbb{R}^n$. The *first minimum* of the lattice, denoted $\lambda_1(\mathscr{L})$, is the length of a shortest nonzero vector $v_1 \in \mathscr{L}$. It is well known that for any full rank lattice, $\lambda_1(\mathscr{L})$ satisfies the Minkowski bound, i.e., $\lambda_1(\mathscr{L}) \leq \sqrt{n}(\det(\mathscr{L}))^{1/n}$. The *second minimum* of the lattice, denoted by $\lambda_2(\mathscr{L})$, is the smallest real number r such that there exist two $\mathbb{R}$-linearly independent lattice vectors v_1 and v_2 such that $\|v_1\|, \|v_2\| \leq r$. In general, for $i = 1, 2, \ldots, m$, the i^{th} *successive minimum* of the lattice, denoted $\lambda_i(\mathscr{L})$, is the smallest real number r such that there exist "i" $\mathbb{R}$-linearly independent vectors $v_1, v_2, \ldots, v_i \in \mathscr{L}$ such that $\|v_1\|, \|v_2\|, \ldots, \|v_i\| \leq r$. If we can find lattice vectors $v_1, v_2, \ldots, v_m \in \mathscr{L}$ which form a basis for $\mathscr{L}$, such that $\|v_i\| = \lambda_i(\mathscr{L})$ for $i = 1, 2, \ldots, m$, then, this basis is almost orthogonal and a "best possible" basis for the lattice. However, such a basis is very difficult to find.

2.2 Permutations and groups

The symmetric group on $X = \{1, 2, \ldots, n\}$ is the collection of all $n!$ permutations of the symbols of X, denoted by Σ_n. Σ_n forms a group under the usual composition of functions. If $\pi \in \Sigma_n$, one of several ways of representing π is as a two-row matrix of the form $\pi = \left(\begin{smallmatrix} 1 & 2 & \cdots & n \\ \pi(1) & \pi(2) & \cdots & \pi(n) \end{smallmatrix}\right)$, exhibiting the bijection. The *Cartesian form* of π is simply the vector of images $[\pi(1), \pi(2), \ldots, \pi(n)]$.

For example $[4,5,3,2,1]$ is the Cartesian form of permutation $\begin{pmatrix}1&2&3&4&5\\4&5&3&2&1\end{pmatrix}$. The symmetric group Σ_n can be made into a metric space by defining the distance between two permutations $x,y\in\Sigma_n$ as the number of places the Cartesian forms of x and y differ. For example, the distance between the permutations $x=[4,5,3,2,1]$ and $y=[4,1,3,2,5]$ is 2. We denote the distance between x and y by $d(x,y)$. Note that this metric is the well known *Hamming distance* on Σ_n.
Given a permutation $\pi\in\Sigma_n$, we associate with π an $n\times n$ matrix $A=(a_{i,j})^T$, where $a_{i,j}=1$ if $\pi(i)=j$ and $a_{i,j}=0$ otherwise. Such a matrix A is called a permutation matrix. This is a matrix obtained from the identity matrix I_n by permuting its rows by π. The $n\times n$ matrix associated with $[4,5,3,2,1]$ is

$$\pi=\begin{pmatrix}0&0&0&0&1\\0&0&0&1&0\\0&0&1&0&0\\1&0&0&0&0\\0&1&0&0&0\end{pmatrix}$$

Obviously, there are $n!$ $n\times n$ permutation matrices. If some of the 1's in A are replaced by -1, the resulting matrix is called a *signed* permutation matrix. Clearly, there are $2^n n!$ signed permutation unimodular matrices that can rearrange the given lattice basis vectors. Let $c\in\Sigma_n$, and r a non negative integer, we define the *sphere* of radius r and center c as follows:

$$S_c(r)=\{x\in\Sigma_n : d(c,x)=r\}.$$

We define the *radius* of a permutation as its *Hamming distance* from the identity permutation. A permutation π_r of radius r in the symmetric group Σ_n is said to be a *left permutation* if $r\le\frac{n}{2}$, otherwise, it is a *right permutation*.

2.3 The Gram-Schmidt Orthogonalization (GSO)

The GSO procedure is a method for getting an orthogonal basis for a given subspace $V\le\mathbb{R}^n$, when a basis B for V is given. Let $B=\{b_1,b_2,\ldots,b_m\}$ be a basis of V. The GSO of B is the basis $B^*=\{b_1^*,b_2^*,\ldots,b_m^*\}$ such that

1. $b_1{}^* = b_1$
2. $b_i{}^* = b_i - \sum_{j=1}^{i-1}\mu_{i,j}b_j^*, \quad$ for $2\le i\le m$.

where,

3. $\mu_{i,j}=\dfrac{\langle b_i,b_j^*\rangle}{\langle b_j^*,b_j^*\rangle}$ for $1\le j<i\le m$) . (1)

From this, we can write $b_i=\sum_{j=1}^{i}\mu_{i,j}b_j^*$ for any $1\le i\le m$. That is $B=MB^*$, Where $M=(\mu_{i,j})$, $\mu_{i,j}=0$ for $i<j$, and $\mu_{i,i}=1$ for $1\le i\le m$. That is,

$$\begin{bmatrix}b_1\\b_2\\\vdots\\b_m\end{bmatrix}=\begin{bmatrix}\mu_{1,1}&0&0&\ldots&0\\\mu_{2,1}&\mu_{2,2}&0&\ldots&0\\\vdots&&\ddots&&\vdots\\\mu_{m,1}&\mu_{m,2}&\mu_{m,3}&\ldots&\mu_{m,m}\end{bmatrix}\begin{bmatrix}b_1^*\\b_2^*\\\vdots\\b_m^*\end{bmatrix}$$

Here, $M=(\mu_{i,j})$ is a lower triangular matrix with $\det(M)=1$. If the vectors $b_1,b_2,\ldots,b_m$ are in $\mathbb{Q}^n$, then so are the vectors $b_1^*,b_2^*,\cdots,b_m^*$. In general, $b_1^*,b_2^*,\ldots,b_m^*$ are not integral linear combinations of the $b_1,b_2,\ldots,b_m$. So, the Gram-Schmidt basis vectors $b_1^*,b_2^*,\ldots,b_m^*$ are not necessarily in the lattice generated by $b_1,b_2,\ldots,b_m$. Here, B^* is an orthogonal basis for the vector subspace $V\le\mathbb{R}^n$, and $\|b_i^*\|\ne 0$ for $1\le i\le m$.

2.4 LLL Reduced Bases

Let $\mathscr{L}=\langle B\rangle_{\mathbb{Z}}$ with basis $B=\{b_1,b_2,...,b_n\}$ be a full rank lattice in $\mathbb{R}^n$ with GSO vectors $\{b_1^*,b_2^*,...,b_n^*\}$. For a real number $\alpha\in\left(\frac{1}{4},1\right)$, B is said to be α-*reduced* (or α-LLL-reduced) if the following conditions hold:

1. $|\mu_{i,j}|\le\frac{1}{2}$ for $1\le j<i\le n$,
2. $\|b_i^*+\mu_{i,(i-1)}b_{i-1}^*\|^2\ge\alpha\|b_{i-1}^*\|^2$, for $2\le i\le n$.

Condition (2) is called the *exchange* condition. We can rewrite (2) as

3. $\|b_i^*\|^2\ge(\alpha-\mu_{i,(i-1)}^2)\|b_{i-1}^*\|^2$, for $2\le i\le n$.

In 1982, A.K. Lenstra, H.W. Lenstra and L. Lovász [25] described an algorithm, known as the LLL algorithm, which produces an $\alpha-$reduced basis B' for the lattice $\mathscr{L}(B)$ in polynomial time. LLL also provides a relatively short lattice vector of length ℓ, and it is convenient to denote this LLL operation in the form: $(B',\ell)\leftarrow\Lambda_\alpha(B)$ or simply, $(B',\ell)\leftarrow\Lambda(B)$. Note that B could be just a

Doi:

generating set of the lattice and not necessarily a basis. However, the resulting B' will always be a basis for $\mathscr{L}(B)$. We will not describe the LLL algorithm here, however, we will be using LLL as a basic *black box* operational component in the basis reduction systems that we will be discussing. For high dimensional lattices it is very hard to find a short vector satisfying the Minkowski bound, using a single LLL application. For practical purposes we consider a lattice reduction procedure satisfactory if the shortest vector $\|v\| \leq m^t(\det(\mathscr{L}))^{1/m}$ where $t \leq 1$, and extremely satisfactory if $t \leq 1/2$.

3 Sensitivity of LLL under basis permutations

We denote a single LLL operation on a permuted basis B^π by $\Lambda(B^\pi)$. In our experiments the row vectors of the basis B of a lattice $\mathscr{L}$ are permuted under the permutation π. A repeated application of LLL on the same ordered basis B yields identical output. Consider a case where a single LLL operation can not completely reduce a lattice basis B. If we apply LLL lattice basis reduction after randomly (or systematically) reordering the original basis B, it will almost always yield a different basis.

Consider a basis $B = B_{54}$, of an ideal lattice of rank 54, taken from the ideal lattice challenge [27]. The length l of the shortest vector of $\Gamma = \Lambda(B)$ turns out to be

$$l = 2745.73.$$

For each radius $r \in \{5, 10, \ldots, 50\}$, we select a random sample P_r of $N = 100$ permutations and compute $\{\Lambda(\Gamma^{\pi_i}) : \pi_i \in P_r\}$. This yields a 100 shortest vectors of lengths $L_r = \{l_1, l_2, \cdots, l_{100}\}$, see table (1) . We then record the minimum, maximum, mean (μ), the standard deviation (σ), and the range of L_r.

radius	min	max	μ	σ	range
05	1946.87	2512.98	2241.26	115.51	566.11
10	1968.17	2535.28	2256.89	102.84	567.11
15	1946.77	2532.94	2256.88	122.84	586.17
20	1946.87	2521.33	2282.65	95.86	574.46
25	1946.87	2469.95	2264.28	96.69	523.08
30	1968.17	2473.17	2263.23	101.65	505.00
35	1968.17	2558.02	2271.41	125.97	589.85
40	1946.77	2493.44	2254.79	115.24	546.67
45	1968.17	2552.18	2284.52	109.11	584.01
50	2010.59	2490.98	2260.74	100.35	480.39

Table 1. LLL reduction over permuted basis

3.1 Effectiveness of right permutation

The significant values of σ and range in table (1) provide strong evidence of the sensitivity of LLL to permuting the lattice basis. We proceed to analyze the reduction pattern when basis rows are permuted. In the experiments below, we take a basis B of an ideal lattice of rank 52 from the *Hall of Fame* website [27] and reduce it by the LLL algorithm. Let $\Gamma = \Lambda(B)$. Γ is not of high quality and requires further reduction. We proceed as follows: for each possible radius $r \in \{2, 3, \ldots, 52\}$, we select a random sample $P_r = \{\pi_{r,1}, \ldots, \pi_{r,N}\}$ of N permutations, where $N \in \{100, 500\}$. Recall that a permutation $\pi_{r,j} \in P_r$ is of distance r from the identity permutation. We compute $\Lambda(\Gamma^\pi)$ for all $\pi \in P = \bigcup_{r=2}^{52} P_r$, and observe how often a basis gets reduced for each sample. The output parameters l, L, and $\log(wt)$ are the resulting lengths of the **shortest** vector, **longest** vector, and the $\log_{10}$ of the **weight** of the reduced basis respectively.

In the graph below, we use the following notation.

$$llb \leftarrow l_{\min}, \qquad lub \leftarrow L_{\min}, \qquad \text{and} \qquad mwt \leftarrow \min(\log_{10} wt)$$

where min represents the minimum over the complete sample of size N, for each radius.

The relative frequency 0.6 corresponding to the radius r in the above graphs represents that the basis gets further reduced 60% of the time if we rearrange Γ with permutations of radius r. From the above graphs, we observe that increasing the permutation radius results in an increase of the relative frequency of the shortest vector. More precisely, the lim_{inf} and lim_{sup} of the relative frequency is monotonically increasing with radius.

Next, we compute the averages for $l, L, \log_{10}(wt)$ over the 500 permutations for each radius $r \in \{2, 3, \ldots, 52\}$ and analyze the pattern of these averages versus the radius. Further, we normalize the average values by using the transformation

$$\text{normalized value} = \frac{\text{value - minimum value}}{\text{maximum value - minimum value}}$$

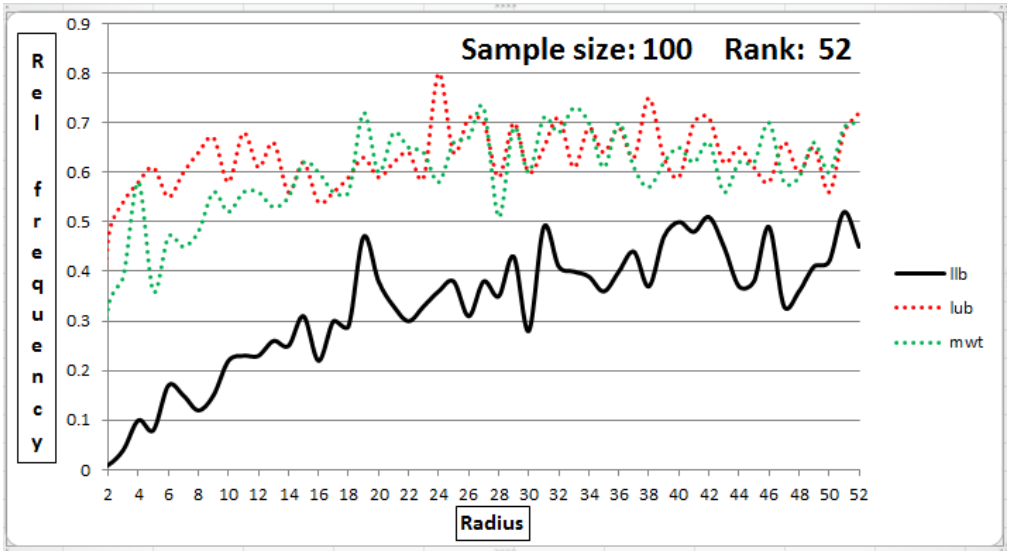


Figure 1. Relative Frequency vs radius

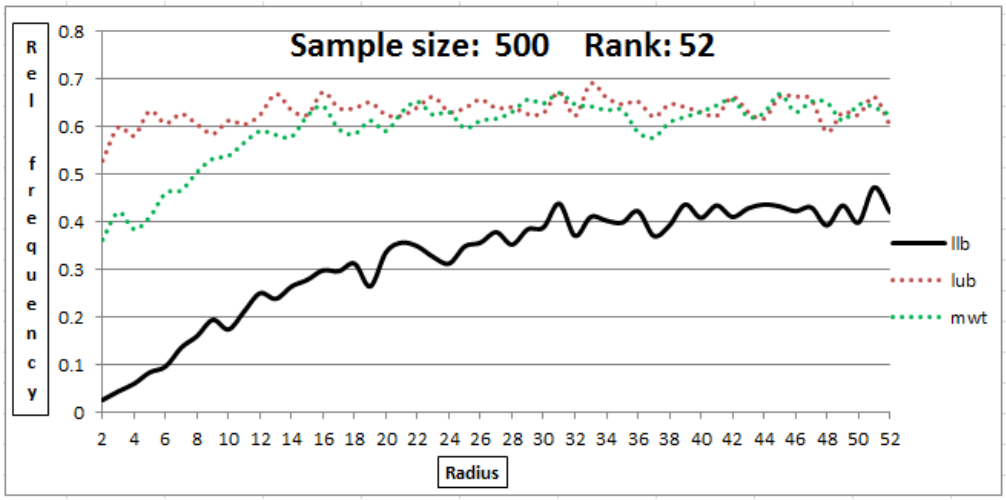


Figure 2. Relative Frequency vs radius

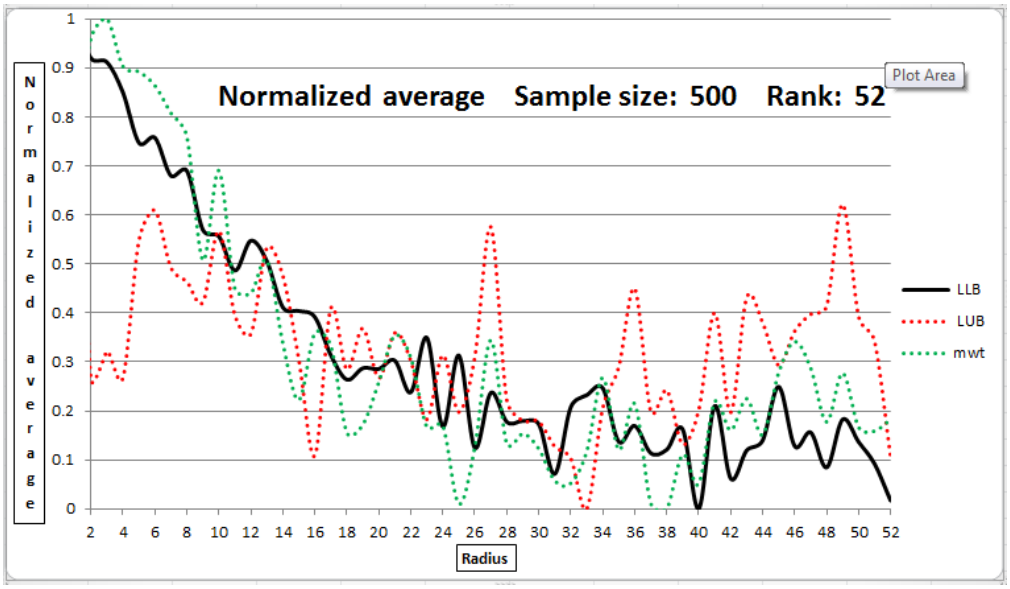


Figure 3. Normalized average vs radius

 doi:

The normalized average values 0 and 1 corresponding to the radii r_1 and r_2 represent the least and the greatest average among all experiments occurring under permutations of radii r_1 and r_2 respectively.
In the above experiments, the decreasing pattern of the curve corresponding to the shortest vector, makes clear that on average, the shortest vector length is achieved more efficiently if the radius of the permutation is larger. This means that the length of the shortest vector is effectively reduced by an application of **right** permutations. This is true for the predominant number of experiments we have performed on lattice bases of different dimensions. Based on our observations in 3.1 and 3.2, we propose the following lattice basis reduction techniques.

4 Hill Climbing Lattice Basis Reduction

Hill climbing reduction is an optimization technique that attempts to search for a better solution locally and repeats the procedure until it finds a prescribed global solution. In our research, this method takes advantage of the sensitivity of LLL to permuting the input basis. We propose two types of Hill climbing lattice basis reduction techniques:

1. Reduction with a fixed radius (spherical random walk).
2. Reduction with variable radius (spiral random walk).

The second method differs from the first in the sense that the radii of successive permutations samples are systematically increased from one step to the next step, while radii are fixed in the first method. In the methods discussed below we relax the upper bound condition from the Minkowski bound ($\lambda_1(\mathscr{L}) \leq m^{1/2} \cdot (\det \mathscr{L})^{1/m}$) to an approximation $\|v\| \leq m \cdot (\det \mathscr{L})^{1/m}$ which is much easier to achieve, but was deemed sufficient for a submission of a solution to the *Ideal Lattice Hall of Fame* [27].

4.1 Hill climbing lattice basis reduction algorithm

(Type I: *Fixed radius*)
Input: Basis $B = \{b_1, b_2, \cdots, b_{m \leq n}\}$ of $\mathscr{L} \subset \mathbb{R}^n$, and parameters: $k,\ p,\ r$.
Output: $v \in \mathscr{L}$, where $\|v\| \leq m \cdot (\det \mathscr{L})^{1/m}$.

1. $l_{min} \leftarrow nmax,\ \pi_{min} \leftarrow identity$
2. for $i \in \{1, 2, \cdots, p\}$ do:
3. for $j \in \{1, 2, \cdots, k\}$ do:
   3.1 Choose a random $\pi \in \Sigma_m$ of radius r
   3.2 $(B', l) \leftarrow \Lambda(B^{\pi})$
   3.3 if $(l \leq l_{min})$ do: $\{l_{min} \leftarrow l \ \ \& \ \ \pi_{min} \leftarrow \pi\}$
   3.4 next j
4. Update: $B \leftarrow B^{\pi_{min}}$
5. next i

(Type II: *Variable radius*)
Input: Basis $B = \{b_1, b_2, \cdots, b_{m \leq n}\}$ of $\mathscr{L} \subset \mathbb{R}^n$ and $r_0, rstep, k, p$, parameters.
Output: $v \in \mathscr{L}$, where $\|v\| \leq m \cdot (\det \mathscr{L})^{1/m}$.

1. $r \leftarrow r_0 - rstep,\ l_{min} \leftarrow nmax,\ \pi_{min} \leftarrow identity$
2. for $i \in \{1, 2, \cdots, p\}$ do:
3. $r \leftarrow r + rstep$
4. for $j \in \{1, 2, \cdots, k\}$ do:
   3.1 Choose a random $\pi \in \Sigma_m$ of radius r
   3.2 $(B', l) \leftarrow \Lambda(B^{\pi})$
   3.3 if $(l \leq l_{min})$ do: $\{l_{min} \leftarrow l \ \ \& \ \ \pi_{min} \leftarrow \pi\}$
   3.4 next j
5. Update: $B \leftarrow B^{\pi_{min}}$
6. next i

doi:

In the tables below, we compare the length of the shortest vector by using the hill climbing reduction approach on the ideal lattice of rank 68 and the ideal lattice of rank 108 taken from the ideal lattice *Hall of Fame* website [27]. The value at HC-r represents the length of the shortest vector achieved using the hill climbing algorithm of radius r. The value at $HC-PSL2$ represents the length of the shortest vector achieved using the hill climbing algorithm in the group $PSL(2,q)$ on $q+1$ points, $q \in \{67, 107\}$. For these examples the LLL-RR (arbitrary precision floating point) version of LLL algorithm from the NTL library [34] is used to reduce the lattice basis. The sample size taken in each step is $k = 100$.

Initially, the length of the shortest vector obtained for the lattice of rank 68 after an application of the LLL-RR algorithm with reduction parameter $\alpha = 0.9$ was 4664. The shortest vector in the experiment in the table below corresponds to the reduction parameter $\alpha = 0.9999$. We have compared the hill climbing reduction with different radii using the same parameter $\alpha = 0.9999$. The length of the shortest vector was 2222 that corresponds to BKZ-10. Here, BKZ-10 represents the Block Korkin Zolotarev [31] lattice basis reduction for block size 10. The tables below contain values from random hill steps.

Step	LLL	BKZ-10	HC-30	HC-42	HC-48	HC-54	HC-66	HC-PSL2
1	2890	2222	2172	2218	2172	2189	2154	2255
5	–	–	2152	2172	2172	2172	**2141**	2182

Table 2. Comparison for an ideal lattice of rank 68

Next, the length of the shortest vector obtained for the lattice of rank 108 after the application of the LLL-RR algorithm with reduction parameter $\alpha = 0.9$ was 12,308. Since the shortest vector corresponds to the LLL parameter $\alpha = 0.99999$, we compare the hill climbing reduction with different radii with the same parameter $\alpha = 0.99999$.

Step	LLL	BKZ-35	HC-50	HC-60	HC-70	HC-80	HC-100	HC-PSL2
1	6275	4616	4258	4129	4309	**3835**	4202	4513
4	–	–	4015	3938	4010	3835	4104	4337
7	–	–	4015	3880	4010	3835	4006	4064

Table 3. Comparison for an ideal lattice of rank 108

From the above tables, we observe that the hill climbing algorithm with any radius in the experiments produces smaller l values than the LLL or BKZ algorithms. Among all values, HC-66 in table (2) and HC-80 in table (3) have better approximations for the shortest lattice vectors. In the next section we observe the hill climbing lattice basis reduction results versus the radius of permutations. We observed a number of experimental results of lattice basis reduction using our hill climbing approach on bases each of which are already reduced by a single LLL operation. In this section, we observe convergence patterns as follows:

(i) for the length of the *shortest lattice vector at each hillstep* versus *radius*.

(ii) *average length and the shortest vectors at all hill steps* versus *radius*.

(iii) *average length* and *shortest length of all radii* versus *each hill step*.

For each step and each possible radius, we have taken a basis sample of 100 permutations. Based on our computations, we conclude that the run-time to get the shortest lattice vector using hill climbing with variable radii is **less** than the approach with fixed radius. Scatterplot with a smooth fitted lines of our experiments are given below.

5 Hill climbing reduction using a cluster

This method reduces the lattice bases using parallel computation. We perform the hill climbing lattice basis reduction algorithm sending the basis B_i to all available computer nodes, along with a single permutation $\pi_{i,j}$ from the random sample S to each node. Clearly this process reduces the computing time from T to process all k LLL operations on a single node, to T/k for each step, where k is the size of the sample of permutation which is also the number of nodes.
Sending a permuted lattice basis to parallel nodes is not a good idea for a lattice basis that requires long runtime. Instead, we "*diffuse*" the given input basis into smaller blocks (not necessarily disjoint), say $\{B_1, B_2, \cdots B_s\}$ with $B = \bigcup_{j=1}^{s} B_j$. Subsequently, we send the B_j to the server nodes with proper instructions to obtain reduced bases B'_j of $\mathscr{L}(B_j)$. The server nodes then send the B'_j to the master node where they are fused to make a new basis B' of the lattice. This method not only reduces the runtime but also reduces the value of the largest entries in the basis. As we get the output back from the server machines to the master client machine, we combine them and rearrange the basis vectors using a **right** permutation. Then, we again diffuse the basis gradually increasing the block size. The process continues until a desired bound is obtained. Using this approach, we have successfully achieved the lattice vectors whose estimated euclidean norm were less than the given bound on the **Lattice hall of fame** [27].

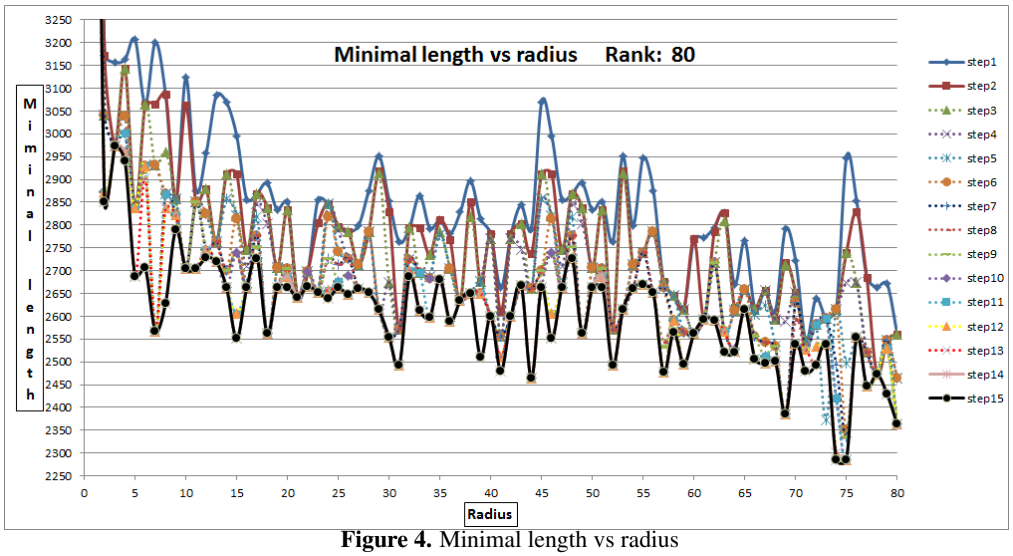


Figure 4. Minimal length vs radius

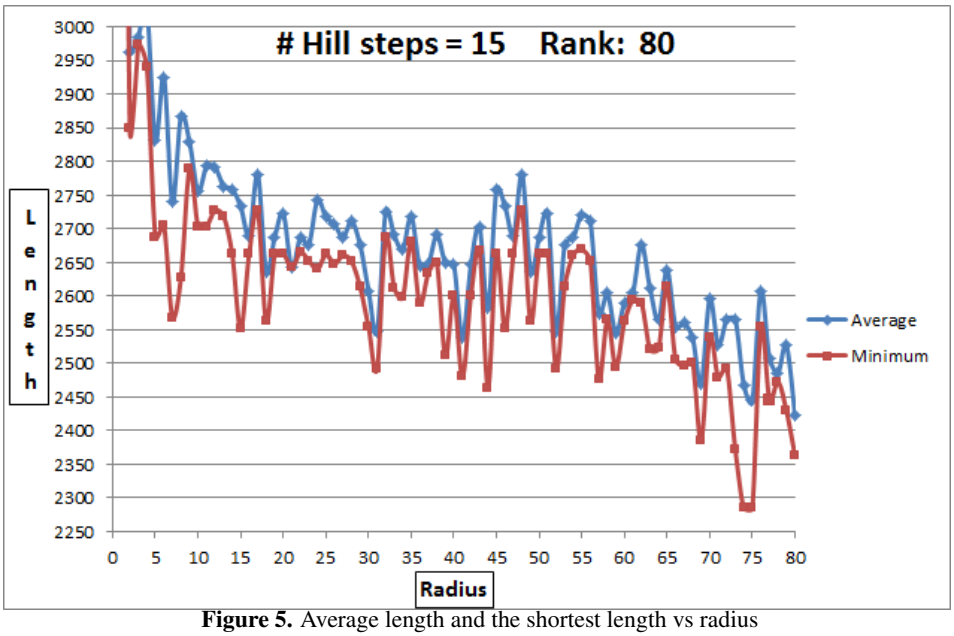


Figure 5. Average length and the shortest length vs radius

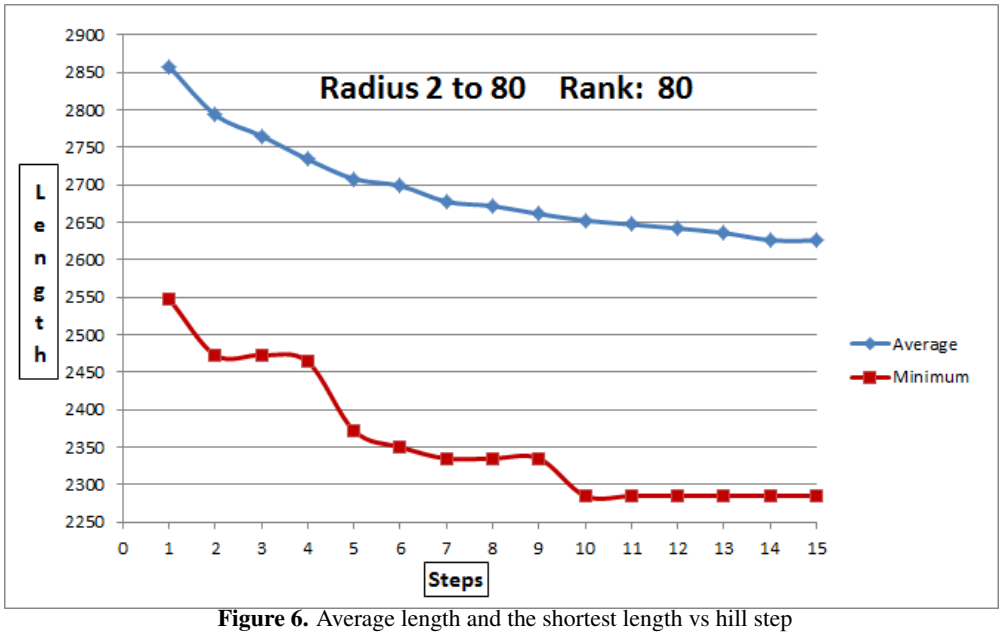


Figure 6. Average length and the shortest length vs hill step

 doi:

5.1 Lattice Diffusion and Sublattice Fusion (LDSF) Algorithm

Input: Basis B of $\mathscr{L} \subset \mathbb{R}^n$, β : Block Size, and b, N, M : parameters.

Output: A lattice vector of "minimal" length close to b.

1. $m \leftarrow$ number of rows in B $\diamond$ $l \leftarrow 1$.
2. $i \leftarrow 1$.
3. For $j = 1$ to k; k is the number of servers.
4. $B_j \leftarrow$ a block matrix of β rows taken randomly from $B \subset \bigcup_{j=1}^{k} B_j$.
5. Send B_j to server j $\diamond$ $B'_j \leftarrow$ result from server j.
6. $(B, \ell) \leftarrow (\Lambda(\cup B'_j))^{\pi}$ $\diamond$ go to step 3 if $M \geq i \leftarrow i+1$.
7. End if $\ell \leq b$.
8. Go to the step 2 if $(N \geq l \leftarrow l+1) \wedge (1 \leq k \leftarrow k-1)$.

The subroutines of lattice diffusion and sublattice fusion (LDSF) algorithms are explained by the figures below. Here LDSF(t, B) represents the algorithm that diffuses the basis B into t blocks sends those blocks into m servers and applies LLL reduction. Finally, the algorithm combines all reduced blocks to get a basis for the lattice.

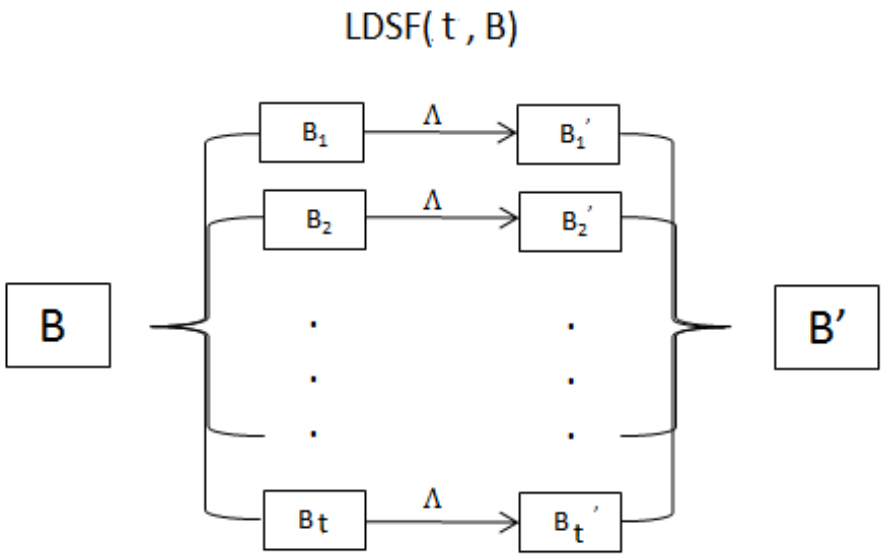


Figure 7. LDSF subroutine algorithm

We presently consider a second operator which we denote by $C = \Sigma(t,k,B)$ which consists of (i) applying a sample of k permutations $\pi_1, \pi_2, \ldots, \pi_k$ to the input base B, (ii) computing $C_i = LDSF(t, B^{\pi_i})$, for $1 \leq i \leq k$, and (iii) letting C be the best basis among the resulting C_i. We say that a basis B of a lattice is better than a basis C of the lattice if the length of the shortest vector in B is less than the length of the shortest vector in C.
If the time required to reduce a basis is very large, we use our multistage lattice diffusion and sublattice fusion technique until we get significant changes of the basis entries. In fact, a combination of the lattice diffusion and the sublattice fusion algorithms with the hill climbing algorithm improves a lattice basis. An explanation of multistage lattice diffusion and sublattice fusion plus a hill climbing algorithm is given below:

5.2 A 4-stage hybrid LDSF-HC process

In the first stage, given an initial basis B_0, which is not satisfactorily reduced by LLL, we compute $LDSF(t, B_0)$ by breaking B_0 into m blocks of sub-bases $B_{0,1}, B_{0,2}, \ldots, B_{0,t}$, and applying the LLL-RR algorithm to each $B_{0,i}$, to obtain $\Lambda(B_{0,1}), \ldots, \Lambda(B_{0,t})$. We then combine the $\Lambda(B_{0,i})$ to obtain basis B_1 of $\mathscr{L}$. This step helps to reduce the absolute value of the lattice basis entries. In the second stage, we obtain a random sample of k right permutations $\pi_1, \ldots, \pi_k$ and compute bases $C_j = LDSF(t, B_1^{\pi_j})$, for $1 \leq j \leq k$. Next, we select the best basis among the $\{C_j\}_{j=1}^{k}$ and call it B_2. The operations performed in stage 2 can be succinctly denoted by $B_2 = \Sigma(t,k,B_1)$. Stage 3 is simply $B_3 = \Sigma(l,k,B_2)$ where $l < t$. Finally there is a termination stage computing the basis $B_4 = \Lambda(B_3)$. Schematically,

 doi:

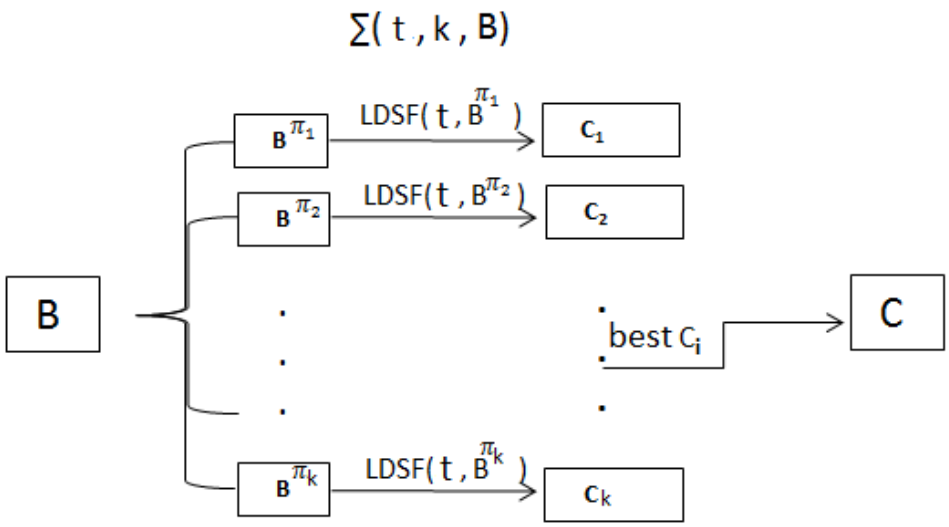


Figure 8. Operator $\Sigma(t,k,B)$

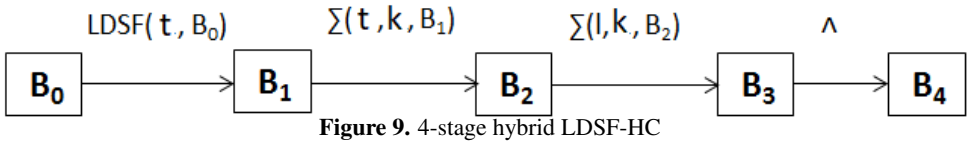


Figure 9. 4-stage hybrid LDSF-HC

If the lattice basis is not sufficiently reduced to use LLL in the 3rd stage, we add additional stages like stage 3 hoping for a better reduction. For example, a 5-stage hybrid LDSF-HC model experiment is given below.
In the experiments below, we compare our lattice diffusion and sublattice fusion algorithm with the fastest floating point version of LLL lattice reduction algorithm (LLL-FP) available in the NTL library [34] with the parameter $\alpha = 0.9$.

5.3 Experiment

The initial information is an ideal lattice of rank 300 obtained from [27].
i) The length of each basis vector is $\approx 10^{900}$.
ii) The initial time to reduce the lattice basis with our device of 24 GB/3GHz i7 processor using the NTL version of LLL-RR, with reduction parameter $\alpha = 0.9$, is approximately a week.
The total runtime by using 5-stage hybrid LDSF-HC is tabulated below.

Stage	# Sample	#*Blocks*	*llb*	*lub*	Runtime
1	1	3	19324936602	8659772341291	1hr 1 m
2	10	6	4509832663	62270235928	3 m 29 sec
3	5	3	2799147462	81977150923	33 m 51 sec
4	5	2	397155	1249483	53 m 40 sec
5	5	2	140973	236873	1 hr 6 m

Table 4. Reduction on an ideal lattice basis of order 300

In the above experiment, the length of a shortest vector reduced from $\approx 10^{900}$ to 140973 in approximately 3.5 hours. Thus the time using this model reduced from 7 days to 3.5 hours.

6 Conclusions

The LLL algorithm is an idempotent operator. If an application of LLL does not reduce satisfactorily an input basis, then a repeated application of LLL on the basis does not change the basis entries, i.e., $\Lambda^2(B) = \Lambda(B)$. However, a permutation of the basis followed by an LLL application will normally generate a different result, i.e., if $\pi \in \Sigma_m$, then almost always $\Lambda(B^\pi) \neq \Lambda(B)$. Thus, LLL is sensitive to permutations of the input basis. Exploiting this sensitivity is at the core of our research. During the experiments discussed above, we observed that the application of right permutations (2.2) on the input basis B has significantly higher probability of getting B further reduced than the application of left permutations (2.2).

Our *hill climbing* (4) algorithm outperforms traditional basis reduction techniques for a basis that is not sufficiently reduced by LLL or BKZ. M. Braun, A. Kohnert, P. Östergard, and A. Wasserman [4], constructed the world's first large set of geometric 2-designs, namely an LS[3][2,3,2^8], invariant under a Singer subgroup in $GL_8(2)$. We have constructed an additional 9 distinct, in fact pairwise non-isomorphic, large sets LS[3][2,3,2^8] [20], with the help of our hill climbing lattice basis-reduction algorithm. Further, using the hill climbing method, we are able to reduce many ideal lattices and publish vectors of several challenge problems in the lattice hall of fame [27].

When the dimension and basis entries increases, hill climbing takes much longer time to reduce the lattice basis. Therefore, a more advanced approach is required. Reducing the basis of a sublattice of smaller rank has a significantly shorter runtime and the resulting sublattice basis has much smaller entries. This property motivates our *lattice diffusion - sublattice fusion* (LDSF) method (5.1). A significant advantage of LDSF method is that it speeds up the process by reducing the magnitudes of the basis entries in the sublattices. In addition to that, each subbase gets reduced faster, and a shortest vector in the sublattice also lies in the lattice. Dividing the lattice basis in k blocks, also called subbases, reduces the LLL reduction algorithm time by an overall factor of approximately k^4. By combining hill climbing and LDSF, we construct a complex hybrid system (5.2) achieving extremely good reduction. This method performed better than currently available algorithms for lattices that had taken longer runtime.

Author Declarations

Author Contributions: Conceptualization, B.K.K.; methodology, B.K.K.; software, B.K.K.; investigation, B.K.K. and S.S.M.; formal analysis, B.K.K.; writing—original draft preparation, B.K.K.; writing—review and editing, B.K.K. and S.S.M. All authors have read and approved the final version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data supporting the findings of this study will be shared upon reasonable request.

Code Availability Statement: All scripts used in this study will be shared upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Institutional Review Board Statement: Not applicable. This study did not involve human participants or animals.

Informed Consent Statement: Not applicable.

Generative AI Disclosure: The authors declare that generative artificial intelligence technologies were not used in the preparation of this manuscript.

References

[1] M. Ajtai, C. Dwork, *A public-Key Cryptosystem with Worst-Case/Average-Case Equivalence*, In Proc. 29th ACM STOC, pp. 284-293, 1997.

[2] M. Ajtai, R. Kumar, and D. Sivakumar, *A Sieve Algorithm for the Shortest Lattice Vector Problem*, In Proc. of STOC, 2001, pp. 601–610.

[3] L. Babai, *On Lovász lattice reduction and the nearest lattice point problem*, Combinatorica, Vol. 6 (1986), pp. 1–13.

[4] M. Braun, A. Kohnert, P. Östergard, A. Vardy, A. Wassermann, *Large Sets of t-designs over Finite Fields*, JCTA, Vol. 124 (2014) pp. 195-202.

[5] J.-Y. Cai and T. W. Cusick, *A Lattice-Based Public-Key Cryptosystem*, Information and Computation, Vol. 151, Issues 1-2, 1999, pp. 17-31.

[6] D. Coppersmith, *Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities*, Journal of Cryptology, 1997, Vol. 10, pp. 233-260.

[7] D. Coppersmith, Adi Shamir, *Lattice Attacks on NTRU*, Advances in Cryptology-EUROCRYPT '97, Vol. 1233 of LNCS, PP. 52-61, Springer 2001.

[8] O. Dagdelen, M. Schneider, *Parallel enumeration of shortest lattice vectors*, In Euro-Par 2010, Vol. 6272 of LNCS, pp. 211-222, Springer, 2010.

[9] C. F. Gauss, *Disquisitiones arithmeticae*, 1801.

[10] C. Gentry, *Key Recovery and Message Attacks on NTRU-Composite*, Advances in Cryptology-EUROCRYPT 2001, Vol. 2045 of LNCS pp. 182-194.

[11] C. GENTRY, *A Fully Homomorphic Encryption Scheme*, Ph.D. thesis, Stanford University, Department of Computer Science, 2009.

[12] O. GOLDREICH, S. GOLDWASSER, S. HALEVI, *Public-Key Cryptosystem from Lattice Reduction Problems*, Advances in Cryptology-CRYPTO 1997, Vol. 1294 of LNCS pp. 112-131.

[13] N. GAMA, P. Q. NGUYEN, *Predicting lattice Reduction*, Eurocrypt 2008: Advances in Cryptology, 31-51.

[14] GUILLAUME HANROT, DAMIEN STEHLÉ* *Worst-Case Hermite-Korkin-Zolotarev Reduced Lattice Bases* Théme SYM-Systémes symboliques Projects Cacao et Arénaire, Rapport de recherche n^{o}6422-Novembre 2007-25 pages.

[15] T, GUNEYSU, V. LYUBASHEVSKY, T. POPPELMANN, *Practical Lattice-Based Cryptography: A signature Scheme for Embedded System*, CHES 2012, LNCS 7428, pp. 530-547, 2012.

[16] J. HASTAD and J. C. LAGARIAS, Math. Annalen 2287 (1990), pp. 167-174.

[17] C. HERMITE, Extraits de lettres de M. Hermite à M. Jacobi sur différents objects de la théorie des nombres, J. Reine Angew. Math., 40 (1850), pp. 279-290.

[18] J. HOFFSTEIN , J. PIPHER , AND J. H. SILVERMAN, *NTRU: A Ring Based Public Key Cryptosystem*, In Proc. of ANTS, 1998, pp.267-288.

[19] J. HOWE, T. PÖPPELMANN, M. O'NEILL, E. O'SULLIVAN, T. GÜNEYSU, *Practical Lattice-Based Digital Signature Schemes*, ACM Trans. Embeded Computing System, 2015, Vol. 14(3), Article No. 41, pp. 1-24.

[20] M. HURLEY, B. KHADKA, and S. S. MAGLIVERAS, *Some new large sets of geometric designs of type LS[3][2,3,2^8]*, Algebra Comb. Discrete Appl. 3(3) (2016), *pp*. 165 – 176

[21] A. KORKIN, and G. ZOLOTAREFF, *Sur lef formes quadratiques positives*, Mathematische Annalen Vol. 6 (1873), Issue 3, pp. 366-389.

[22] J. C. LAGARIAS and A. M. ODLYZKO, *Solving low-density subset sum problems*, J. Assoc. Comp. Mach., 32 (1985), pp. 229-246.

[23] J. C. LAGARIAS, H. W. LENSTRA, JR. and C. P. SCHNORR, *Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice*, Combinatorica, 10 (4) (1990), pp. 333-348.

[24] L. LAGRANGE, *Recherches d'arithmétique,*Nouv. Mém. Acad., Berlin, 1773.

[25] A. K. LENSTRA, H. W. LENSTRA JR., and L. LOVÁSZ, *Factoring Polynomials with Rational Coefficients*, Mathematische Annalen, 261 (1982), pp. 515-534.

[26] H. MINKOWSKI, *Geometrie der Zahlen*, Leipzig und Berlin, Druck und Verlag von B. G. Teubner, 1910.

[27] T. PLANTARD, F. GÖPFERT, *Ideal Lattice Challenge* http://latticechallenge.org/ideallattice-challenge/

[28] M. SEYSEN, *Simultaneous reduction of a lattice basis and its reciprocal basis*, Combinatorica, September 1993, Volume 13, Issue 3, pp 363-376.

[29] M. SCHNEIDER and J. BUCHMANN, *Extended Lattice Reduction Experiments using the BKZ algorithm*, Sicherheit 2010, LNI 170, pp. 241-252, GI, 2010.

[30] C. P. SCHNORR, *A hierarchy of polynomial time lattice basis reduction algorithms*, Theoretical Computer Science, Volume 53, Issues 2-3, (1987), pp. 201–224.

[31] C.P. SCHNORR, *Block Korkin-Zolotarev Bases and Successive Minima*, *citeseerx.ist.psu.edu/*, 1992.

[32] C. P. SCHNORR, M. EUCHNER, *Lattice basis reduction: Improved practical algorithm and solving subset sum problems*. Mathematical Programming, August 1994, vol 66, pp. 181-199.

[33] A. SHAMIR, *A polynomial time algorithm for breaking the Merkle-Hellman cryptosystem*, In Proc. of the 23rd IEEE Symposium on the Foundations of Computer Science, 1982, pp. 145-152.

[34] V. SHOUP, *Number Theory Library (NTL)*, http://www.shoup.net/ntl/

[35] B. L. VAN DER WARDEN, *Die Reduktionstheorie Der Positiven Quadratischen Formen*, Acta Mathematica, December 1956, Volume 96, Issue 1, pp. 265-309.

Publisher's Note. Statements, opinions, and data contained in publications are solely those of the individual author(s) and contributor(s). The publisher and editors disclaim responsibility for injury to people or property resulting from ideas, methods, instructions, or products referred to in the content.