Vol.2013-CSEC-62 No.13
Vol.2013-SPT-6 No.13
2013/7/18

格子の最短ベクトル問題における探索空間の特定

深瀬 道晴[1,a)]　柏原 賢二[2,b)]

概要：格子暗号の安全性は，格子における最短ベクトルを求める問題 (SVP) に密接に関連している．通常，最短ベクトルを求めるためには，格子の基底簡約におけるステップを数多く繰り返し，徐々に短いベクトルを求める必要がある．本研究においては，ある程度簡約が進んだ基底に関して，最短ベクトルの存在する探索空間を推測し，一度のステップで最短ベクトルを求める手法を提案する．また，一つの基底に関する探索空間において探索する場合と比べて，基底を変化させることで，探索空間を変化させながら探索した場合の方が，最短ベクトルが求まる頻度が高いことを実験的に示す．

1. はじめに

整数格子は，線形独立な整数成分を持つベクトルの組である基底によって定義される．本稿では，格子というとき整数格子のことを指すこととする．格子は，基底のベクトルの整数線形結合の全ての集合である．いくつかの格子の問題は計算困難であることが知られている．例えば，格子における最短ベクトルを求める問題 (SVP) が計算困難であることが知られている．最短ベクトルとは、原点以外の原点に最も近い点を表すベクトルのことである．

計算困難であるいくつかの格子の問題に基づいて，公開鍵暗号システムが提案されている．格子の問題に基づく公開鍵暗号システムは，一般に格子暗号と呼ばれている．格子暗号の有用な特徴については，([8]) 等を参照されたい．

格子暗号の安全性評価には，格子基底簡約アルゴリズムが用いられてきた ([1], [3])．格子基底簡約とは，与えられた格子の基底から短いベクトルからなる基底を求めることである．簡約された基底の中の一番短いベクトルが、最短ベクトルとなることがある．代表的な格子基底簡約アルゴリズムとして，LLL アルゴリズム [5] が知られている．その他の基底簡約アルゴリズムとして，BKZ アルゴリズム [10] や RSR アルゴリズム [11] 等が知られている．これらは，LLL アルゴリズムを拡張したアルゴリズムであり，LLL アルゴリズムより長い計算時間を要するが，より短い基底を出力する．

BKZ アルゴリズムなどの基底簡約アルゴリズムにおいては，通常，最短ベクトルを求めるために多くのステップを要する．例えば，BKZ アルゴリズムにおいては，LLL アルゴリズムと ENUM[10] を繰り返すことで，基底のベクトルを徐々に短くする必要がある．このとき，高次元においては，BKZ アルゴリズムは非常に長い計算時間を要する．本研究においては，BKZ アルゴリズムなどによる簡約がある程度進んだ基底に対して，最短ベクトルの存在する探索空間を推測し，一度のステップで非常に短い基底ベクトルを求める手法を提案する．

最短ベクトルを含む探索空間を推測するために, 基底ベクトルには属していない既知の短いベクトルの情報を利用する．基底行列が与えられると、既知の短いベクトルに対する表現が決まる．基底行列を変化させたときの、その既知の短いベクトルに対する表現の分布を調べることで、最短ベクトルがどのあたりに存在しやすいのかの情報を推測することができる．その情報に基づいて，探索空間を決定する．

探索空間を定義する際に，RSR アルゴリズムのアイデアを拡張した．最短ベクトルを一度のステップで求めるためには，RSR アルゴリズムにおいて用いられている探索空間を拡張する必要がある．本研究では，拡張した探索空間を Extended Search Space(ESS) とよぶ．

ESS は，基底をパラメータの一つとする．そのため，最短ベクトルが ESS に含まれるかは基底に依存する．ある ESS が最短ベクトルを含まない場合，基底を除くパラメータが同一であっても，基底を変化させれば，最短ベクトルを含むことがある．本研究では，このことを利用して，ESS のパラメータの基底を変化させながら探索することで最短ベクトルが求まる頻度を高めることができることを実験的に示す．

[1] 獨協大学
Dokkyo University
[2] 東京大学
The Uniersity of Tokyo
[a)] fukase@dokkyo.ac.jp
[b)] kashiwa@idea.c.u-tokyo.ac.jp

最近，格子基底簡約アルゴリズムなどの有効性を測る指標として，Darmstadt 工科大学の Lattice challenge や SVP challenge が注目されている ([6]，[9])．提案手法の有効性を示すために，ESS における探索を SVP challenge のいくつかの問題に適用した．その結果，次元 96，98，100 の SVP において最短ベクトルを効率的に求めることができた．

2. 格子

2.1 格子と基底簡約アルゴリズム

格子 $L \subset \mathbb{R}^m$ は,n 個の線形独立なベクトル列 $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \mathbb{Z}^{m \times n}$ によって，$L(B) = \{B\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^n\}$ のように定義される．整数 n を，L の次元という．$n = m$ のとき，L は全次元であるという．ベクトル列 $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \mathbb{Z}^{m \times n}$ を，L の基底という．基底ベクトルとは，基底行列の列ベクトルのことをいう．基底ベクトルの順序は，格子基底簡約において重要であり，基底ベクトルの順序が異なれば，異なる基底と考える．本稿では，正方行列の基底 $B \in \mathbb{Z}^{n \times n}$ によって生成される全次元整数格子 $L \subset \mathbb{Z}^n$ のみを扱う．$n \geq 2$ のとき，同一の格子を生成する基底は無数に存在する．

L における最短ベクトル，すなわち，$\mathbf{0}$ を除いて最も短いベクトルのノルムを $\lambda_1(L)$ によって表す．ある格子の基底 B が与えられたとき，$L(B)$ の最短ベクトルを求める問題を最短ベクトル問題 *(SVP)* という．長さが $\lambda_1(L)$ に等しい，または，それに近い格子ベクトルを求めるための最も一般的な方法は，格子基底簡約である．格子基底簡約は，格子の任意の基底から短いベクトルからなる基底を求めることである．低い次元では、簡約された基底の中の一番短いベクトルが、最短ベクトルとなることがある．格子基底簡約アルゴリズムとして，LLL アルゴリズムと BKZ アルゴリズムが良く知られている．LLL アルゴリズムは，多項式時間内に理論的に保証された短さの基底を出力する．BKZ アルゴリズムは，LLL アルゴリズムと全探索アルゴリズム ENUM を組み合わせたものである．BKZ アルゴリズムは，最近 Chen 等によって BKZ2.0 として改良された [1]．BKZ アルゴリズムや RSR アルゴリズムのような格子基底簡約アルゴリズムは，サブルーチンとして LLL アルゴリズムを含んでいる．LLL アルゴリズムにおいて重要な過程が，*Gram-Schmidt* 直交化法である．Gram-Schmidt 直交化法によって，格子の基底 B から *Gram-Schmidt* 直交ベクトル列が計算される．Gram-Schmidt 直交ベクトル列はベクトル空間 R^n を生成する基底となるが，一般に格子の基底とはならない．格子の基底 $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ について，基底のベクトルが互いに直交に近い程，基底のベクトルが短くなることが知られている [7]．Gram-Schmidt 直交化法により得られる直交ベクトル列は有理数係数による演算で得られるため，一般に格子ベクトルにはならない．そのため，LLL アルゴリズムでは，Gram-Schmidt 直交化法の各ステップにおいて，直交ベクトル列をそれらに近い格子ベクトルで代替する．これによって，LLL アルゴリズムは，直交に近い格子の基底を生成する．

基底 $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ に対して，Gram-Schmidt 直交ベクトル列 $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^* \in \mathbb{R}^n$ が次のように定義される：$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j}\mathbf{b}_j^* \quad \text{with} \quad \mu_{i,j} = \langle \mathbf{b}_i, \mathbf{b}_j^* \rangle / \langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle$．ここで，$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i$ は，$\mathbb{R}^n$ における内積である．全ての i について，$\mathbf{b}_i^*$ は，$\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$ に直交する $\mathbf{b}_i$ の成分である．したがって，$\mathbf{b}_i^*$ と $\mathbf{b}_j^* (j \neq i)$ とは直交する．

2.2 Gram-Schmidt 係数の 012 表現

本研究においては，Random Sampling Reduction(RSR) アルゴリズムのアイデアにおけるように，格子ベクトルを，基底 B の Gram-Schmidt 直交ベクトル列を用いて表現する．$\mathbf{x} \in \mathbb{Z}^n$ として，$\mathbf{v} = B\mathbf{x}$ を，基底 B によって生成される格子における格子ベクトルとする．ここで，$\mathbf{v}$ を，Gram-Schmidt 直交ベクトル列 $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ と B の Gram-Schmidt 直交化の係数 $\mu_{i,j}$ とを用いて，$\mathbf{v} = \sum_{j=1}^{n} \nu_j \mathbf{b}_j^*$ のように表せる．$\boldsymbol{\nu} \in \mathbb{R}^n$ は，それぞれの j について $\nu_j = \sum_{i=1}^{n} x_i \mu_{i,j}$ である．$\mathbf{b}_j^*$ は互いに直交するため，$\|\mathbf{v}\|^2 = \sum_{j=1}^{n} \nu_j^2 \|\mathbf{b}_j^*\|^2$ である．以下，ν_j を $\mathbf{v}$ の *Gram-Schmidt* 係数という．

$\mathbf{v}$ の Gram-Schmidt 係数 ν_j は，$\mathbf{v}$ と Gram-Schmidt 直交ベクトル列 $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ がどれだけ直交に近いかを表している．LLL アルゴリズムによる簡約基底は，直交に近い基底であり，全ての基底ベクトルの ν_j は，一つの j を除いて 0.5 以下である．このように，$\nu_j \leq 0.5$ のとき，直交に近いという意味で j に 0 を対応させる．

Schnorr は，$\mathbf{v}$ の ν_j について，$\nu_j \leq 0.5$ のものだけでなく，$0.5 < \nu_j \leq 1$ のものを考えることによって，RSR アルゴリズムを導入した ([11])．このように，$0.5 < \nu_j \leq 1$ のとき，j に 1 を対応させる．

著者等は，以下のことを確認した ([2])．基底 B が BKZ アルゴリズム等によって簡約されている場合，最短ベクトルの ν_j の大部分は 0 に対応し，それ以外はほとんど 1 に対応する．そして，時々 $1 < \nu_j \leq 1.5$，または，$1.5 < \nu_j \leq 2$ となる．このように，$1 < \nu_j \leq 1.5$ のとき j に 2 を対応させ，$1.5 < \nu_j \leq 2$ のとき j に 3 を対応させる．

このような表現を Gram-Schmidt 係数の 012 表現と呼ぶ．BKZ アルゴリズムのサブルーチン ENUM においても，内部的に 012 表現が用いられている．

3. 提案手法

3.1 ESS

本研究では, ([2]) で示した最短ベクトルの Gram-Schmidt 係数の分布に基づいて，最短ベクトル，または，それに近い格子ベクトルを求めるための探索空間 Extended Search Space(ESS) を提案する．著者等は，[2] において ESS とい

う概念を既に定義したが，本研究において精緻に定義し直す．

格子の基底を B，B の Gram-Schmidt 直交ベクトル列を $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$, $L(B)$ の任意の格子ベクトルを，$\mathbf{v} = \sum_{j=1}^{n} \nu_j \mathbf{b}_j^*$ とする．ここで，$\boldsymbol{\nu} \in \mathbb{R}^n$ である．また，$\mathbf{v} = \sum_{j=1}^{n} \nu_j \mathbf{b}_j^*$ に対して，$\mathbf{z}(\mathbf{v}) \in \mathbb{Z}^n$ s.t. $z_j(\mathbf{v}) = \lfloor 2|\nu_j| \rceil$ とする．ある $k \in \mathbb{N}$ について $d_k(\mathbf{v}) = \#\{z_j(\mathbf{v}) : z_j(\mathbf{v}) = k, 1 \leq j \leq n\}$, $w_k(\mathbf{v}) = n - \min\{j : z_j(\mathbf{v}) = k\} + 1$，ある $c \in \mathbb{Z}_+$ について $\mathbf{s} \in \mathbb{N}^c$,$\mathbf{t} \in \mathbb{N}^c$ とする．例えば，$d_1(\mathbf{v})$ は，$\mathbf{v}$ の Gram-Schmidt 係数の 012 表現における 1 の個数を表す．ここで，集合 S に対して，$\#S$ は S の要素数を表す．このとき，格子ベクトルの集合 ESS $V_B(\mathbf{s}, \mathbf{t})$ を以下のように定義する．

定義 1 $V_B(\mathbf{s}, \mathbf{t}) = \{\mathbf{v} \in L(B) : \mathbf{d}(\mathbf{v}) \leq \mathbf{s}, \mathbf{w}(\mathbf{v}) \leq \mathbf{t}\}$.

ここで，ベクトルに関する不等号は，各成分が小さいことを表す．

3.2 ESS のパラメータの推測

$V_B(\mathbf{s}, \mathbf{t})$ は，$\mathbf{s}$ によって Gram-Schmidt 係数の 012 表現における 0 や 1 等の個数を制限し，$\mathbf{t}$ によって 0 や 1 等が存在する範囲を規定している．2 節で述べたように，基底 B が BKZ アルゴリズム等によって簡約されていれば，最短ベクトルの Gram-Schmidt 係数の 012 表現として，大部分が 0 に対応し，それ以外はほとんどが 1，時々 2 か 3 に対応する．よって，$c = 3$ 程度を考えることにする．そのため，定義 1 のように ESS $V_B(\mathbf{s}, \mathbf{t})$ を定義したとき，最短ベクトルの Gram-Schmidt 係数の 012 表現の傾向に基づいてパラメータ $\mathbf{s}, \mathbf{t}$ を決定すれば，$V_B(\mathbf{s}, \mathbf{t})$ は高確率で最短ベクトルを含む．そして，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含むとき，$V_B(\mathbf{s}, \mathbf{t})$ に含まれる格子ベクトルを全て調べる，すなわち，網羅列挙すれば，最短ベクトルが求まることになる．$V_B(\mathbf{s}, \mathbf{t})$ のサイズ，すなわち，$\#V_B(\mathbf{s}, \mathbf{t})$ は，基底 B によらず，$\mathbf{s}, \mathbf{t}$ のみから計算できる．

最短ベクトルが ESS $V_B(\mathbf{s}, \mathbf{t})$ に含まれるかは，パラメータの一つ基底 B に依存する．そのため，$\mathbf{s}, \mathbf{t}$ が同一であっても，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含むかどうかは異なる．本研究では，このことを利用して，基底 B を変化させながら探索することで最短ベクトルが求まる頻度を高めることができることを実験的に示す．

$V_B(\mathbf{s}, \mathbf{t})$ の最適なパラメータ $\mathbf{s}, \mathbf{t}$ は，既知の短い格子ベクトルの情報を利用することで推測できる．ここで，既知の短い格子ベクトルとは，既に答えが分かっている格子基底簡約問題から得られる情報である．具体的に，情報とは，その SVP における格子の最短ベクトル，または，それに近い格子ベクトルの Gram-Schmidt 係数の 012 表現の分布である．この分布は，特定の格子によらないと考えられる．このため，答えの分かっていない格子基底簡約問題に適用できるというのが本研究の立場である．

4. 実験結果

3 節において説明した ESS を用いた提案手法を，SVP challenge の SVP に適用する．SVP challenge は，World Wide Web 上で公開されている Darmstadt 工科大学の SVP の問題である．SVP challenge では，格子の次元毎に SVP が設定されている．現在解かれている SVP の中で，最高次元のものは次元 126 の SVP である．次元 126 の SVP も含めて既に解かれた SVP の多くは，現在最も有効なアルゴリズムと考えられる BKZ アルゴリズムとそのサブルーチン ENUM，また，BKZ アルゴリズムの改良型である BKZ2.0 によって解かれている．また，本研究で用いた全てのプログラムコードは Java で書かれており，全てのプログラムは 3.4GHz クアッドコア Intel Core i7 の iMac 上で実行されている．

本研究では，3 節で述べた 2 つのアイデアに基づいて，ESS の網羅列挙を用いて SVP challenge の SVP を解く．1 つ目は，既に解かれている SVP の最短ベクトルの情報から，ESS $V_B(\mathbf{s}, \mathbf{t})$ のパラメータ $\mathbf{s}, \mathbf{t}$ を調整することである．2 つ目は，ESS $V_B(\mathbf{s}, \mathbf{t})$ のパラメータの一つである基底 B 自体を繰り返し変化させることで，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度を高くすることである．以下，それぞれのアイデアの詳細を述べる．

4.1 探索空間の特定

ここで，既に解かれている SVP の最短ベクトルの情報を用いることを考える．まず，既に求められている最短ベクトルから，その最短ベクトルを含む $V_B(\mathbf{s}, \mathbf{t})$ を逆算することができる．

例えば，次元 98 のある SVP の解 $\mathbf{v}$ は，$\mathbf{v} = (\ -217\ -394\ \ 365\ 329\ \ -378\ \ -153\ \ -55\ \ 370\ \ -268\ \ -288\ 118\ \ -114\ \ -410\ \ -53\ \ -153\ \ -815\ \ 74\ \ -159\ \ 344\ -172\ \ 301\ 24\ \ -22\ \ -183\ \ 163\ 319\ 199\ \ -236\ \ 516\ -475\ \ 237\ \ -88\ \ 239\ \ -170\ \ 185\ \ -21\ \ -41\ \ -395\ -378\ \ 187\ 74\ \ -193\ \ -10\ \ 233\ 511\ \ -286\ \ -304\ \ 401\ -256\ \ -330\ \ -203\ \ 69\ 56\ \ -281\ \ 62\ \ -439\ \ 29\ \ -93\ 272\ \ -28\ \ -9\ \ 57\ 237\ 257\ \ -24\ \ -224\ \ 485\ \ -259\ 77\ 30\ 288\ 13\ \ -465\ \ 142\ 35\ 230\ 229\ 44\ 27\ \ -126\ \ -221\ 246\ \ -32\ \ 155\ \ -133\ \ -64\ \ 135\ 85\ \ -445\ \ 72\ 37\ \ -323\ 314\ \ -139\ \ -210\ \ 317\ \ -259\ \ -314\ )$ である．ここで，$\mathbf{v}$ を対応する格子の $(0.99, 29)$-BKZ 簡約基底 B の Gram-Schmidt 直交ベクトル列 $\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*$ によって表す．このとき，Gram-Schmidt 係数 $\boldsymbol{\nu} = (\ -0.02143\ \ -0.04457\ -0.03453\ \ 0.01593\ 0.09816\ 0.05755\ \ -0.1042\ \ -0.1233\ 0.03286\ \ -0.0964\ \ -0.1395\ \ -0.2312\ \ -0.0593\ \ 0.1591\ -0.02132\ \ 0.07724\ \ -0.07\ \ -0.1298\ \ -0.06475\ \ -0.1372\ 0.00375\ \ -0.07147\ \ -0.06738\ \ -0.09508\ \ 0.1833$

−0.1051 0.0139 0.05985 0.09455 0.05835 0.201 0.02585 −0.2853 0.00266 −0.287 −0.06112 0.01072 −0.1282 −0.3068 −0.04872 0.00203 −0.1983 −0.1188 0.1108 0.1463 0.08779 −0.04489 −0.0578 −0.145 −0.1426 0.3933 0.5958 0.4872 0.4078 0.175 −0.5278 0.05106 −0.06086 −0.0136 −0.2115 −0.06433 0.4199 0.3314 −0.3249 −0.7389 −0.1159 0.1048 −0.1967 −0.06134 −0.3449 −0.4379 −0.2301 0.2042 0.2968 0.2141 −1.04 −0.4057 −1.019 −0.145 0.9208 0.2747 −0.02767 0.7106 −0.621 −0.16 0.6967 −0.125 0.2919 1.156 −0.08425 0.5864 0.06202 1.592 0.8476 −0.7954 −0.9682 −0.5077 −1) となる．

上の $\mathbf{v}$ に対して, $\mathbf{z} =$ (0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 2 0 2 0 1 0 0 1 1 0 1 0 0 2 0 1 0 3 1 1 1 1 1) が Gram-Schmidt 係数の 012 表現である．

上の $\mathbf{v}$ に対して，$d_0(\mathbf{z}) = 81$，$d_1(\mathbf{z}) = 13$，$d_2(\mathbf{z}) = 3$, $d_3(\mathbf{z}) = 1$, $w_0(\mathbf{z}) = 92$, $w_1(\mathbf{z}) = 47$, $w_2(\mathbf{z}) = 23$, $w_3(\mathbf{z}) = 6$ となる．定義 1 から，$\mathbf{v}$ は $V_B((81, 13, 3, 1), (92, 47, 23, 6))$ に含まれる．したがって，$V_B((81, 13, 3, 1), (92, 47, 23, 6))$ に含まれる格子ベクトルを網羅列挙すれば, 次元 98 の SVP の解 $\mathbf{v}$ は求まることになる．既知の最短ベクトルを改めて求めることには意味はないが，このようにして逆算して求めた ESS $\mathbf{s}, \mathbf{t}$ は未知の最短ベクトルを求める際に有用な情報となる．

4.2 基底の交換

ここでは，ESS $V_B(\mathbf{s}, \mathbf{t})$ のパラメータの一つである基底 B を繰り返し変化させることを考える．B が異なっても $\mathbf{s}, \mathbf{t}$ が同一であれば，$V_B(\mathbf{s}, \mathbf{t})$ のサイズ，すなわち，$\#V_B(\mathbf{s}, \mathbf{t})$ は変わらない．しかし，サイズは同一であっても，ある B の $V_B(\mathbf{s}, \mathbf{t})$ は最短ベクトルを含んでいるが，ある B' の $V_{B'}(\mathbf{s}, \mathbf{t})$ は最短ベクトルを含んでいないということが起こる．それは，基底が異なれば，基底の Gram-Schmidt 直交ベクトル列も異なるため，最短ベクトルの Gram-Schmidt 係数の 012 表現も異なるためである．このことを利用して，B を繰り返し変化させて，小さなサイズの $V_B(\mathbf{s}, \mathbf{t})$ を数多く生成し，いずれかの $V_B(\mathbf{s}, \mathbf{t})$ において最短ベクトルを求めるという戦略を取ることができる．ここで特に主張したいことは，ある一つの基底 B の大きなサイズの $V_B(\mathbf{s}, \mathbf{t})$ において探索するよりも，基底 B を繰り返し変化させながら比較的小さなサイズの $V_B(\mathbf{s}, \mathbf{t})$ において探索する方が効率良く最短ベクトルを求めることができるということである．

上の主張を支持する実験結果を，以下に示す．ここでは，次元 100 の SVP について，基底 B を 3750 回交換しながら，3750 個の ESS $V_B(\mathbf{s}, \mathbf{t})$ を生成した．そして，それぞれの $V_B(\mathbf{s}, \mathbf{t})$ に最短ベクトルが含まれるかどうかという判定を行い，最短ベクトルが含まれる頻度を計算した．ここで，頻度は，最短ベクトルを含む $V_B(\mathbf{s}, \mathbf{t})$ の個数を $\#V_B(\mathbf{s}, \mathbf{t})$ で割ったものである．これを，パラメータ $\mathbf{s}, \mathbf{t}$ を変化させながら計算した．以下の実験においては，パラメータ $\mathbf{s} = (90, 8, 2), \mathbf{t} = (100, 35, 6)$ を基点に $\mathbf{s}, \mathbf{t}$ のいずれかの成分を変化させた．

図 1 は，$\mathbf{s} = (90, 8, 2), \mathbf{t} = (100, 35, 6)$ について，s_3，及び，$\mathbf{t}$ を固定，$s_2 = 8$ の値を変化させながら (それに伴い s_1 の値も変化)，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度を計算した結果を示している．図 1 においては，横軸の s_2 の値の増加に従い，$\#V_B(\mathbf{s}, \mathbf{t})$ が大きくなるが，$s_2 = 8$ をピークに縦軸の頻度は減少していくことが分かる．このことは，ESS のサイズを一定以上に大きくすることが非効率であることを示している．

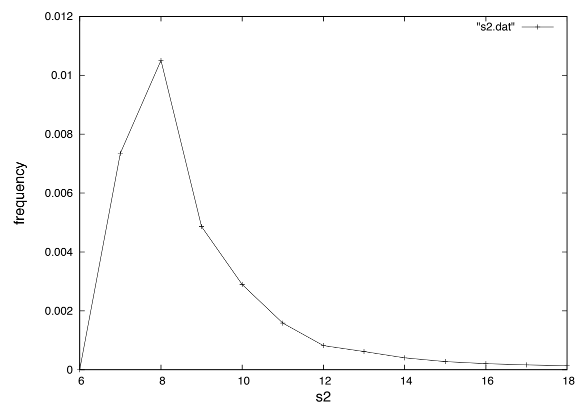

図 1 次元 100 において，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度．横軸が s_2 の値，縦軸が頻度を表している．

図 2 は，$\mathbf{s} = (90, 8, 2), \mathbf{t} = (100, 35, 6)$ について，s_2，及び，$\mathbf{t}$ を固定，$s_3 = 2$ の値を変化させながら (それに伴い s_1 の値も変化)，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度を計算した結果を示している．図 2 においては，横軸の s_3 の値の増加に従い，$\#V_B(\mathbf{s}, \mathbf{t})$ が大きくなるが，頻度は増加し

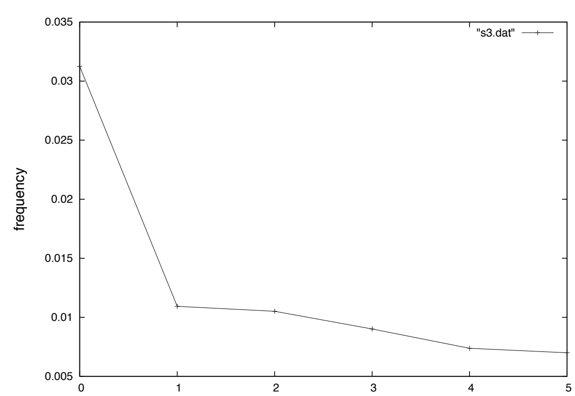

図 2 次元 100 において，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度．横軸が s_3 の値，縦軸が頻度を表している

ていないことが分かる．

図 3 は，$\mathbf{s} = (90, 8, 2), \mathbf{t} = (100, 35, 6)$ について，$\mathbf{s}$，及び，t_1, t_3 を固定，$t_2 = 35$ の値を変化させながら，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度を計算した結果を示している．図 3 においては，横軸の t_2 の値の増加に従い，$\#V_B(\mathbf{s}, \mathbf{t})$

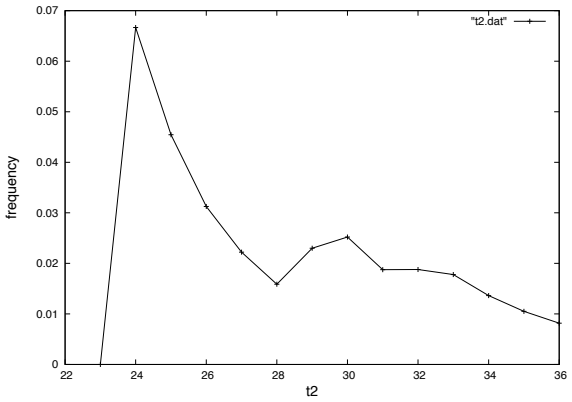

図 3　次元 100 において，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度．横軸が t_2 の値，縦軸が頻度を表している

が大きくなるが，$t_2 = 24$ をピークに縦軸の頻度は減少していくことが分かる．

図 4 は，$\mathbf{s} = (90, 8, 2), \mathbf{t} = (100, 35, 6)$ について，$\mathbf{s}$，及び，t_1, t_2 を固定，$t_3 = 6$ の値を変化させながら，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度を計算した結果を示している．図 4 においては，横軸の t_3 の値の増加に従い，$\#V_B(\mathbf{s}, \mathbf{t})$

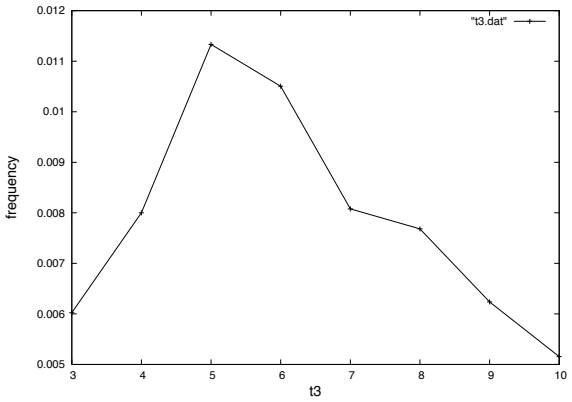

図 4　次元 100 において，$V_B(\mathbf{s}, \mathbf{t})$ が最短ベクトルを含む頻度．横軸が t_3 の値，縦軸が頻度を表している

が大きくなるが，$t_3 = 5$ をピークに縦軸の頻度は減少していくことが分かる．

以上の結果から，上の主張，ある一つの基底 B の大きなサイズの $V_B(\mathbf{s}, \mathbf{t})$ において探索するよりも，基底 B を繰り返し変化させながら比較的小さなサイズの $V_B(\mathbf{s}, \mathbf{t})$ において探索する方が効率良く最短ベクトルを求めることができるということが実験的に示された．

4.3 SVP challenge の求解

本研究では，ESS の網羅列挙を，次元 96，98，100 の SVP に適用した．これらは，既に本研究の手法とは別の手法によって解かれていることを注意しておく．しかし，本研究における実験においては，既に解かれている最短ベクトルの情報は用いておらず，最短ベクトルを除く短ベクトルの情報を用いて SVP を解いた．

SVP を解く際に，本節で述べた 2 つのアイデアを適用した．すなわち，1 つ目のアイデアの探索空間の特定については，最短ベクトルを除く短ベクトルの情報を用いて ESS $V_B(\mathbf{s}, \mathbf{t})$ のパラメータ $\mathbf{s}, \mathbf{t}$ を調整した．2 つ目のアイデアの基底の交換については，$V_B(\mathbf{s}, \mathbf{t})$ のパラメータの 1 つである B を繰り返し変化させ，効率的な探索空間のサイズの $V_B(\mathbf{s}, \mathbf{t})$ において探索を行った．

以上の方針に基づいて SVP の求解を行った結果，次元 96，98，100 のそれぞれにおいて既に解かれている最短ベクトルと同一のベクトルを数日程度で求めることができた．

5. まとめ

本研究では，格子ベクトルの集合である探索空間 ESS を定義した．ある程度簡約の進んだ基底をパラメータとする ESS を用いて，一度のステップで最短ベクトル，または，それに近い格子ベクトルを求めることができることを示した．また，基底を変化させながら探索することで，ESS を変化させ，最短ベクトルが求まる頻度を高めることができることを実験的に示した．今後の展望として，Java によるプログラムコードの C++によるプログラムコードへの変換や計算機環境の整備などを進め，より高次元の SVP に取り組む予定である．

謝辞　本研究の一部は，獨協大学情報学研究所研究助成によるものである．

参考文献

[1] Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better Lattice Security Estimates. Proceedings of ASIACRYPT 2011, vol. 7073 of LNCS, pp. 1-20, (2011).
[2] Fukase, M., Yamaguchi, K.: Exhaustive Search for Finding a Very Short Vector in High-Dimensional Lattices. Proceedings (short papers) of IWSEC 2010, pp. 26-41, (2010).
[3] Gama, N., Nguyen, P.Q.: Predicting Lattice Reduction. EUROCRYPT 2008, vol. 4965 of LNCS, pp. 31-51, (2008).
[4] Gama, N., Nguyen, P.Q., Regev, O.: Lattice Enumeration Using Extreme Pruning. EUROCRYPT 2010, vol. 6110 of LNCS, pp. 257-278, (2010).
[5] Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring Polynomials with Rational Coefficients. Mathematische Ann., vol. 261, pp. 513-534, (1982).
[6] Lindner, R., Rückert, M.: Lattice challenge. Available at `http://www.latticechallenge.org/`.
[7] Micciancio, D.: Improving Lattice Based Cryptosystems Using the Hermite Normal Form. Silverman, pp. 126-145,

(2001).
[8] Micciancio, D: The Geometry of Lattice Cryptogoraphy. Foundations of Security Analysis and Design VI, pp. 185-210, (2012).
[9] Schneider, M., Gama, N.: SVP challenge. Available at `http://www.latticechallenge.org/svp-challenge/`.
[10] Schnorr, C.P., Euchner, M.: Lattice Basis Reduction: Improved Practical Algorithms and Solving Subset Sum Problems. Math. Programming, vol. 66, pp. 181-199, (1994).
[11] Schnorr, C.P.: Lattice Reduction by Random Sampling and Birthday Methods. STACS 2003, vol. 2607 of LNCS, Springer-Verlag, pp. 145-156, (2003).