JAIST Repository
https://dspace.jaist.ac.jp/

Title	Schnorrの素因数分解アルゴリズムの評価
Author(s)	谷, 仁裕
Citation	
Issue Date	2024-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/18888
Rights	
Description	Supervisor: 藤崎 英一郎, 先端科学技術研究科, 修士(情報科学)

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修士論文

Schnorr の素因数分解アルゴリズムの評価

谷 仁裕

主指導教員 藤崎 英一郎

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 6 年 3 月

Abstract

RSA is a popular public key cryptosystem. The security of the RSA relies on the difficulty of prime factoring large composite numbers. So we need to evaluate the size of composite numbers that can be prime factorized. For numbers of the form $N = pq$ (p, q is prime) used in the RSA, the largest number of bits that is currently prime factorized is 829 bits. The algorithm used for this prime factorization is the number field sieve (NFS).

The prime factorization method used in the inside NFS is briefly explained. Suppose x, y are given that satisfy $x^2 \equiv y^2 \pmod{N}$. Then the greatest common divisor $\gcd(x \pm y, N)$ may be a prime factor of N. When prime factors of N are obtained, the prime factorization can be done. To use this prime factorization method, we need to find x, y. In the NFS, we collect several expressions of a certain form (called the factor-relation) and use them to obtain x, y.

There are two types of prime factorization algorithms using lattices proposed by Schnorr that perform prime factorization in a manner similar to the NFS. The algorithm is based on the shortest vector problem (SVP) and the closest vector problem (CVP). Both algorithms are based on computational problems in the lattice. SVP is the problem of finding the shortest vector in a lattice. CVP is the problem of finding a vector in a lattice that is close to a given vector. This paper describe about an algorithm based on SVP. This algorithm constructs a lattice and then collects factor-relations from short vectors in the lattice, and it is stated that it is capable of prime factorizing large numbers. However, the effectiveness of the algorithm has not been fully analyzed at present, so we discuss the effectiveness of the algorithm.

The algorithm based on SVP constructs two values u, v from short vectors in the lattice. If $u - vN$ is small, factor-relation is reliably obtained. Specifically, when the n-th prime number is p_n, if $u - vN \leq p_n$, the factor-relation can be obtained reliably. So assuming that the shortest vector in the lattice was obtained, we obtained the order of computational complexity of the values related to $u - vN$ from the upper bound of the norm of the shortest vector. We then investigated the behavior of $|u - vN|$ when each parameter of the lattice was varied. The results show that $u - vN$ cannot be manipulated to a value smaller than p_n From this result, an algorithm based on SVP cannot reliably obtain factor-relation. So it is thought that difficult to factorize large composite numbers. Also the algorithm was implemented and tested for every 10 bits from a 20-bit composite number and successfully prime factorized a composite numbers up to 50 bits. This result also suggests that it is difficult to prime factorize composite numbers with a large number of bits.

目 次

第 1 章　はじめに **1**

第 2 章　数学的準備 **3**
2.1 格子 3
2.1.1 格子の定義と性質 3
2.1.2 最短ベクトル問題 6
2.2 平方差法による素因数分解 6
2.3 ガンマ関数 7
2.4 大きな素因数を持たない自然数について 8

第 3 章　Schnorr のアルゴリズム **9**
3.1 Schnorr の素因数分解アルゴリズム 9
3.1.1 素因数分解アルゴリズム 9
3.1.2 $u - vN$ について 10

第 4 章　最短ベクトルに対する u-vN **12**
4.1 最短ベクトルに対する u-vN の値 12
4.2 C のみ増加させた場合 12
4.3 n と C を変化させた場合 14

第 5 章　実験結果 **18**

第 6 章　おわりに **20**

表 目 次

5.1 素因数分解実験結果 18

第1章　はじめに

現在広く使われている公開鍵暗号の 1 つである RSA 暗号は大きな合成数の素因数分解が困難であることを安全性の根拠としている．そのため，現実的な時間で素因数分解可能な合成数の大きさを評価するためにこれまで様々な素因数分解アルゴリズムが考えられている．RSA 暗号で使われる $N = pq$ (p, q は素数) の形をした合成数のうち現時点で素因数分解されている最も大きな合成数のビット数は 829 ビット [1] である．このとき用いられたのは最も効率の良いアルゴリズムとして知られている数体篩法 [2] という素因数分解アルゴリズムであり，大きな合成数 N を素因数分解するときの計算量は準指数時間で $O(e^{((\frac{64}{9})^{\frac{1}{3}}+o(1))(\log N)^{\frac{1}{3}}(\log\log N)^{\frac{2}{3}})}$ であることが知られている．

数体篩法は平方差法と呼ばれる方法を用いて大きな合成数 N の素因数分解を行う．平方差法では $x^2 \equiv y^2 \pmod N$ を満たす x, y が与えられたとき，最大公約数 $\gcd(x \pm y, N)$ が N の素因数になる可能性があることを利用して合成数の素因数分解を行う方法である．平方差法を用いるためには x, y を求める必要があり，数体篩法では特定の形をした式 (関係式と呼ぶ) を複数集めそれらの式から x, y を構成する方法をとっている．

数体篩法と同様の流れで素因数分解を行うアルゴリズムとして，Schnorr により提案された格子を用いた 2 種類の素因数分解アルゴリズムがある．最短ベクトル問題 (SVP) を基にしたアルゴリズム [3] と最近ベクトル問題 (CVP) を基にしたアルゴリズム [4] である．どちらのアルゴリズムも格子上の計算問題を基にしており，SVP は格子上の最短なベクトルを見つける問題であり，CVP はあるベクトルが与えられたときにそのベクトルに最も近い格子上のベクトルを見つける問題である．本研究では SVP を基にしたアルゴリズムを扱う．このアルゴリズムは格子を構成したのち格子上の短いベクトルから関係式を集める方法をとっており，大きな合成数を素因数分解可能であると述べられている．しかし現在のところ有効性について十分な解析が行われていないため，アルゴリズムの有効性について考察を行った．

SVP を基にしたアルゴリズムでは格子上の短いベクトルから 2 つのある値 u, v を構成し，$u - vN$ が小さいとき関係式が確実に得られるアルゴリズムとなっている．具体的には n 番目の素数を p_n としたとき $|u - vN|$ が p_n までの素数で素因数分解できれば関係式が得られるため $|u - vN| \leq p_n$ ならば関係式が確実に得られる．そこで，格子上の最短ベクトルが得られたと仮定し，最短ベクトルのノルムの上界から $u - vN$ に関係する値のオーダーを求め，格子の各パラメータを操作したときの $|u - vN|$ のふるまいを調べたところ，$|u - vN|$ を p_n 以下の値に操作する

ことはできないことが分かった．この結果よりSVPを基にしたアルゴリズムでは関係式を確実に得られるとは限らないため大きな合成数の素因数分解はできないといえる．実際にアルゴリズムを実装し20ビットの合成数から10ビットごとに実験を行ったところ最大で50ビットの合成数の素因数分解に成功したが，それより大きいビット数の合成数は素因数分解できなかった．

本論文では，まず2章で数学的準備として格子の定義や性質，平方差法，大きな素因数を持たない数の性質について述べる．3章ではSchnorrの素因数分解アルゴリズムで関係式を集める流れについて説明し，関係式が得られる条件に付いて述べる．4章では格子のパラメータを調整することで関係式が得られる条件を満たすように操作可能か調べる．5章ではアルゴリズムを実装し，実際に素因数分解を行った結果について述べる．

第2章　数学的準備

2.1　格子

2.1.1　格子の定義と性質

ベクトル空間 $\mathbb{R}^m$ の d 個のベクトルを $\mathbf{b}_1, \ldots, \mathbf{b}_d$ とし，これらのベクトルの整数係数の線形結合全体の集合を L とする．

$$L := \left\{ \sum_{i=1}^{d} c_i \mathbf{b}_i \mid c_i \in \mathbb{Z} \right\}$$

d 個のベクトル $\mathbf{b}_1, \ldots, \mathbf{b}_d$ が一次独立であるとき，集合 L を格子と呼び d を次元と呼ぶ．また，格子を生成する d 個のベクトルの組 $\{\mathbf{b}_1, \ldots, \mathbf{b}_d\}$ を格子の基底と呼び，各 $\mathbf{b}_i$ を基底ベクトルと呼ぶことにする．そして基底ベクトル $\mathbf{b}_i$ を行に持つ行列 $\mathbf{B} := (\mathbf{b}_1, \ldots, \mathbf{b}_d)^{\mathrm{T}}$ を基底行列と呼び，基底行列 $\mathbf{B}$ によって生成される格子を $\mathcal{L}(\mathbf{B})$ と表す．また，$\mathrm{span}(L)$ は格子 L の全てのベクトルにより生成される実数ベクトル部分空間とする．

ここからはいくつか格子の性質について述べる．ユニモジュラ行列と呼ばれる行列式が ± 1 の $d \times d$ 整数行列を $\mathbf{T}$ とする．このとき $\mathbf{T}$ を基底行列 $\mathbf{B}$ の左から掛けてできる新たな行列 $\mathbf{TB}$ もまた同じ格子を生成する別の基底を持つ基底行列となる．また $d \geq 2$ のときユニモジュラ行列は無限に存在するため，同じ格子を生成する基底行列は無限に存在する．

任意の基底行列 $\mathbf{B}$ に対して格子の体積を $\mathrm{vol}(L) := \sqrt{\det(\mathbf{B}\mathbf{B}^{\mathrm{T}})}$ とする．$\mathbf{B}$ が正方行列のとき，$\mathrm{vol}(L) = |\det(\mathbf{B})|$ となる．前述したとおり同じ格子を生成する基底は無限個存在するが，格子の体積 $\mathrm{vol}(L)$ は同じ格子を生成する任意の基底に対して不変である．

格子 L 上の最短な非零ベクトルのノルムを $\lambda_1(L)$ とする．このとき d 次元格子 L に対して以下の不等式 (定理 2.1.4) が成り立つ．

$$\lambda_1(L) < \sqrt{d}\mathrm{vol}(L)^{\frac{1}{d}} \tag{2.1}$$

式 (2.1) を示すためにいくつか補題を示す．また以下を定義しておく．

定義 2.1.1.
$\mathbb{R}^m$ の部分集合を S とする．任意の $\mathbf{x} \in S$ に対して $-\mathbf{x} \in S$ であるとき，集合 S は原点に関して対称であるという．また，任意の $\mathbf{x}, \mathbf{y} \in S$ と任意の実数 $0 \leq t \leq 1$ に対して $t\mathbf{x} + (1-t)\mathbf{y} \in S$ であるとき，集合 S は凸であるという．

補題 2.1.2.
格子 L と体積を持つ集合 $S \subseteq \mathrm{span}(L)$ に対して，以下の不等式が成り立つとする．

$$\mathrm{vol}(S) > \mathrm{vol}(L)$$

このとき，2 つの異なる元 $\mathbf{z}_1, \mathbf{z}_2 \in S$ が存在して $\mathbf{z}_1 - \mathbf{z}_2 \in L$ を満たす．

Proof.
格子 L に対するある基底行列 $\mathbf{B}$ を用いて集合 $P_\mathbf{x}$ を以下のようにおく．

$$P_\mathbf{x} = \{\mathbf{y} + \mathbf{x} \mid \mathbf{y} \in \mathcal{P}(\mathbf{B})\}$$
$$\mathcal{P}(\mathbf{B}) = \left\{ \sum_{i=1}^{n} r_i \mathbf{b}_i \mid r_i \in \mathbb{R},\ 0 \leq r_i < 1 \right\}$$

このとき，集合 S を次のように分割する．

$$S = \bigsqcup_{\mathbf{x} \in L} S_\mathbf{x}, \qquad S_\mathbf{x} = S \cap P_\mathbf{x}$$

また，各 $S_\mathbf{x}$ を平行移動させた集合 $S'_\mathbf{x}$ を次のようにおく．

$$S'_\mathbf{x} = S_\mathbf{x} - \mathbf{x} = (S - \mathbf{x}) \cap \mathcal{P}(\mathbf{B})$$

ここで，2 つの異なる $\mathbf{x}, \mathbf{y} \in L$ に対して $S'_\mathbf{x} \cap S'_\mathbf{y} = \emptyset$ ($\emptyset$ は空集合を表す) とすると以下の式が成り立つ．

$$\sum_{\mathbf{x} \in L} \mathrm{vol}(S'_\mathbf{x}) = \mathrm{vol}\left(\bigsqcup_{\mathbf{x} \in L} S'_\mathbf{x} \right) \leq \mathrm{vol}(\mathcal{P}(\mathbf{B}))$$

一方で，仮定 $\mathrm{vol}(S) > \mathrm{vol}(L)$ より以下が成り立つ．

$$\sum_{\mathbf{x} \in L} \mathrm{vol}(S'_\mathbf{x}) = \sum_{\mathbf{x} \in L} \mathrm{vol}(S_\mathbf{x}) = \mathrm{vol}(S) > \mathrm{vol}(L)$$

これは，$\mathrm{vol}(L) = \mathrm{vol}(\mathcal{P}(\mathbf{B}))$ に矛盾する．したがって $S'_\mathbf{x} \cap S'_\mathbf{y} \neq \emptyset$ を満たす異なる $\mathbf{x}, \mathbf{y} \in L$ が存在する．ここで，$\mathbf{z}$ を $S'_\mathbf{x} \cap S'_\mathbf{y} \neq \emptyset$ の元とし，$\mathbf{z}_1, \mathbf{z}_2$ を以下のようにおく．

$$\mathbf{z}_1 = \mathbf{z} + \mathbf{x} \in S_\mathbf{x}, \qquad \mathbf{z}_2 = \mathbf{z} + \mathbf{y} \in S_\mathbf{y}$$

このとき，$\mathbf{z}_1, \mathbf{z}_2$ は異なる S の元である．また $\mathbf{z}_1 - \mathbf{z}_2 = \mathbf{x} - \mathbf{y} \in L$ である． □

補題 2.1.3.
d 次元格子 L と原点に関して対称であり，かつ体積を持つ凸集合 $S \subseteq \mathrm{span}(L)$ に対して，以下の不等式が成り立つとする．

$$\mathrm{vol}(S) > 2^d \mathrm{vol}(L)$$

このとき，S は L 上の非零なベクトルを含む．

Proof.
集合 S' を以下のように置く．

$$S' = \{\mathbf{x} \in \mathbb{R}^m \mid 2\mathbf{x} \in S\}$$

このとき以下の式が成り立つ．

$$\mathrm{vol}(S') = 2^{-d}\mathrm{vol}(S) > \mathrm{vol}(L)$$

補題 2.1.2 より $\mathbf{z}_1 - \mathbf{z}_2 \in L$ となる異なる $\mathbf{z}_1, \mathbf{z}_2 \in S'$ が存在する．S' の定義より，$2\mathbf{z}_1, 2\mathbf{z}_2 \in S$ となり，S の原点対称性より $-2\mathbf{z}_2 \in S$ となる．また S が凸であることより以下の $2\mathbf{z}_1, -2\mathbf{z}_2 \in S$ の中点も S に含まれる．

$$\frac{2\mathbf{z}_1 - 2\mathbf{z}_2}{2} = \mathbf{z}_1 - \mathbf{z}_2$$

したがって，S は非零ベクトル $\mathbf{z} = \mathbf{z}_1 - \mathbf{z}_2 \in L$ を含む． □

定理 2.1.4.
d 次元格子 L に対して以下の不等式が成り立つ．

$$\lambda_1(L) < \sqrt{d}\mathrm{vol}(L)^{\frac{1}{d}}$$

Proof.
格子 L の $\mathrm{span}(L)$ の部分集合を以下のように置く．

$$S = \mathcal{B}(\mathbf{0}, \sqrt{d}\mathrm{vol}(L)^{\frac{1}{d}}) \cap \mathrm{span}(L)$$

ただし $\mathbf{x} \in \mathbb{R}^m$ を中心とした半径 r の開球を $\mathcal{B}(\mathbf{x}, r) = \{\mathbf{z} \in \mathbb{R}^m \mid \|\mathbf{x} - \mathbf{z}\| < r\}$ とする．このとき，S は原点に関して対称な凸集合であり，辺の長さが $2\mathrm{vol}(L)^{\frac{1}{d}}$ の d 次元超立方体を含むため体積は $2^d\mathrm{vol}(L)$ よりも大きい．したがって，補題 2.1.3 より S は L 上の非零ベクトル $\mathbf{z}$ を含むため以下が成り立つ．

$$\lambda_1(L) \leq \|\mathbf{z}\| < \sqrt{d}\mathrm{vol}(L)^{\frac{1}{d}}$$

□

2.1.2 最短ベクトル問題

d次元格子の基底行列$\mathbf{B}$が与えられたとき，以下の式を満たすベクトル$\mathbf{v} \in \mathcal{L}(\mathbf{B})$を見つける問題を最短ベクトル問題 (SVP) と呼ぶ.

$$\|\mathbf{v}\| = \lambda_1(\mathcal{L}(\mathbf{B}))$$

2.2 平方差法による素因数分解

Schnorr の素因数分解アルゴリズムでは合成数を素因数分解するときに平方差法を用いるため平方差法について説明する．2つの異なる奇素数をp, qとし$N = pq$の形の合成数Nを分解したいとする．また素数を昇順にn個並べたものを$p_1, \ldots, p_n$とする．このとき以下の形で表されたそれぞれ異なる式（関係式と呼ぶ）が$n+2$個与えられているとする．($p_0 = -1$とする)

$$\prod_{i=1}^{n} p_i^{e_{i,j}} \equiv \prod_{i=0}^{n} p_i^{e'_{i,j}} \pmod{N} \quad (e_{i,j}, e'_{i,j} \in \mathbb{Z}) \tag{2.2}$$

このとき式 (2.2) は$\prod_{i=0}^{n} p_i^{e_{i,j}-e'_{i,j}} \equiv 1 \pmod{N}$と変形できるため以下の$n+1$次元ベクトルを考える．ただし，$j = 1, \ldots, n+2,\ e_{0,j} = 0$とする．

$$\begin{aligned}\mathbf{v}_j &= (v_{0,j},\ v_{1,j},\ \cdots, v_{n,j}) \\ &= (-e'_{0,j},\ e_{1,j} - e'_{1,j},\ \cdots, e_{n,j} - e'_{n,j})\end{aligned}$$

このとき$n+1$次元ベクトルが$n+2$個あるため以下の式を満たす$t_1, \ldots, t_{n+2} \in \{0, 1\}$を求められる.

$$\sum_{j=1}^{n+2} t_j \mathbf{v}_j \equiv (0, \ldots, 0) \pmod{2}$$

求めた$t_1, \ldots, t_{n+2}$を用いてXを次のようにおく.

$$X = \prod_{i=0}^{n} p_i^{\frac{1}{2}\sum_{j=1}^{n+2} t_j v_{i,j}}$$

このとき以下の式が成り立つ.

$$X^2 \equiv 1 \pmod{N}$$

よって$X^2 - 1$はNの倍数であるため，$(X+1)(X-1)$に分解することで$(X \pm 1)$はNの約数となりpかqとなることが期待できる．最大公約数$\gcd(X \pm 1, N)$を計算することでpかqが得られれば素因数分解ができる．$\gcd(X \pm 1, N)$が1かNになり分解できないときは，別のXを探す．この方法で素因数分解をするためには最初に$n+2$個の関係式を集める必要がある．[1] では格子上の短いベクトルから関係式を構成する方法を提案している.

2.3 ガンマ関数

定義 2.3.1. (ガンマ関数)
$x \in \mathbb{R}$ に対する以下の関数 $\Gamma(x)$ をガンマ関数と呼ぶ.

$$\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$$

ガンマ関数の性質について以下でいくつか示す。

定理 2.3.2.
$x \in \mathbb{N}$ に対して以下の式が成り立つ.

$$\Gamma(x+1) = x!$$

Proof.
ガンマ関数の定義より $\Gamma(x+1)$ は以下のように表せる.

$$\begin{aligned}
\Gamma(x+1) &= \int_0^\infty t^x e^{-t}dt \\
&= \left[-t^x e^{-t}\right]_0^\infty - \int_0^\infty -xt^{x-1}e^{-t}dt \\
&= x\int_0^\infty t^{x-1}e^{-t}dt \\
&= x\Gamma(x) \\
&= x(x-1)(x-2)\cdots 2\Gamma(1) \\
&= x(x-1)(x-2)\cdots 2\int_0^\infty e^{-t}dt \\
&= x!
\end{aligned}$$

□

定理 2.3.3.
$n \in \mathbb{N}$ のとき以下の式が成り立つ. ただし $\sim$ は $n \to \infty$ としたときに左辺と右辺の比が 1 になることを表す.

$$n! \sim n^n e^{-n}$$

Proof.
まず $n!$ の対数をとると以下のように表せる.

$$\log(n!) = \sum_{i=1}^{n} \log i$$

上式の右辺は以下のように表せる．

$$\begin{aligned}\sum_{i=1}^{n} \log i &\sim \int_1^n \log x dx \\ &= [x \log x]_1^n - \int_1^n dx \\ &= n \log n - [x]_1^n \\ &= n \log n - n + 1\end{aligned}$$

よって対数を元に戻すと以下の式が成り立つ．

$$n! \sim n^n e^{-n} \qquad (n \to \infty)$$

□

2.4 大きな素因数を持たない自然数について

定義 2.4.1. (y-smooth)
ある数 $x \in \mathbb{N}$ がもつ最大の素因数が y 以下であるとき，x は y-smooth な数であるという．また，x 以下の y-smooth な数の個数を次のように表す．ただし $M(a)$ は a がもつ最大の素因数を表す．

$$\psi(x, y) = \{1 \leq a \leq x \mid M(a) \leq y\}$$

定義 2.4.2. (dickman function)
次の条件を満たす連続関数 $\rho(u)$ を dickman function と呼ぶ．

$$\begin{cases} \rho(u) = 1 & \text{if } 0 \leq u \leq 1 \\ u\rho'(u) + \rho(u-1) = 0 & \text{if } u > 1 \end{cases}$$

$\psi(x, y)$ と dickman function $\rho(u)$ について y^u が十分大きいとき以下の式が成り立つことが知られている [5]．

$$\frac{\psi(y^u, y)}{y^u} = \rho(u) \tag{2.3}$$

式 (2.3) の左辺は y^u が y-smooth な数となる確率とみることができる．また，$\rho(u)$ に関して以下の不等式が成り立つ [6]．

$$\rho(u) \leq \frac{1}{\Gamma(u+1)} \tag{2.4}$$

式 (2.3) と式 (2.4) より，y^u が y-smooth な数になる確率はガンマ関数 $\Gamma(u+1)$ の逆数で抑えられるため，u が大きくなるほど y^u が y-smooth な数になる確率は小さくなることが分かる．

第3章　Schnorrのアルゴリズム

3.1　Schnorrの素因数分解アルゴリズム

Schnorrの素因数分解アルゴリズムはある基底行列を構成して格子を生成し，その格子上の短いベクトルから関係式を構成し必要な数の関係式が集められたとき平方差法で合成数の素因数分解を行う．ここではSchnorrの素因数分解アルゴリズムで関係式を得るまでの流れについて説明し，関係式が得られるときの条件について述べる．

3.1.1　素因数分解アルゴリズム

ここではアルゴリズムの流れを説明する．まず以下の基底行列$\mathbf{B}$を構成する．ただし$f : [1, 2, \ldots, n] \to [1, 2, \ldots, n]$は置換とし，$C > 0$とする．

$$\mathbf{B} = \begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_n \\ \mathbf{b}_{n+1} \end{pmatrix} = \begin{pmatrix} f(1) & \cdots & 0 & C \ln p_1 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & f(n) & C \ln p_n \\ 0 & \cdots & 0 & C \ln N \end{pmatrix}$$

次にこの基底行列$\mathbf{B}$で生成される格子$\mathcal{L}(\mathbf{B})$上の短いベクトルを求める．格子$\mathcal{L}(\mathbf{B})$上のベクトルは基底$\{\mathbf{b}_1, \ldots, \mathbf{b}_{n+1}\}$の整数係数の線形結合で表せるので$\mathbf{z} \in \mathbb{Z}^{n+1}$としたとき以下のような短いベクトル$\mathbf{zB}$が得られる．

$$\begin{aligned} \mathbf{zB} &= \left(z_1, \cdots, z_{n+1}\right) \begin{pmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{n+1} \end{pmatrix} \\ &= \left(z_1 f(1),\ \cdots\ ,\ z_n f(n),\ \textstyle\sum_{i=1}^{n} (z_i C \ln p_i) + z_{n+1} C \ln N\right) \end{aligned}$$

$\mathbf{zB}$は短いベクトルなので各要素の値は小さい．ベクトルの最初のn個の要素はfが1からnの値なので小さい値となっている．そして最後の要素が小さな値であるということは小さな値をϵと置いたときに最後の要素を以下のように表すことができる．

$$\sum_{i=1}^{n} (z_i C \ln p_i) + z_{n+1} C \ln N = \epsilon$$

N が大きな合成数のとき $C\ln p_i$ よりも $C\ln N$ の方が大きいため，$C\ln p_i$ の整数倍の総和と $C\ln N$ が打ち消し合って小さな値となっているような短いベクトルが得られると考えられる．式で表すと次のようになる．

$$\sum_{i=1}^{n}(z_i C\ln p_i) - C\ln N = \epsilon \tag{3.1}$$

したがって，ここからは $z_{n+1} = -1$ として考えていくことにする．このとき式 (3.1) は以下のように表すことができる．

$$\begin{aligned}&\sum_{i=1}^{n}(z_i C\ln p_i) - C\ln N = \epsilon\\ &\Rightarrow\ \ln p_1^{z_1}\cdots p_n^{z_n} - \ln N = \frac{\epsilon}{C}\end{aligned}$$

ここで，$u = \prod_{z_i>0} p_i^{z_i},\ v = \prod_{z_i<0} p_i^{-z_i}$ とおくと

$$\ln\frac{u}{vN} = \frac{\epsilon}{C}$$

4 節の式 (4.2) より，C よりも ϵ の増加スピードが遅いため C を大きな値でとることで $\frac{\epsilon}{C}$ を 0 に近づけることができる．よって $\frac{u}{vN}$ が 1 に近づくため，$u - vN$ が小さくなると考えることができる．そして u は p_n-smooth な数であるため，$|u - vN|$ が p_n-smooth な数ならば $u, u - vN$ はどちらも p_n-smooth な数となる．そのため次のように合同式をおくことで関係式が得られる．

$$\begin{aligned}u &\equiv u - vN \pmod N\\ \Rightarrow \prod_{i=1}^{n} p_i^{e_{i,j}} &\equiv \prod_{i=0}^{n} p_i^{e'_{i,j}} \pmod N\end{aligned}$$

基底行列 $\mathbf{B}$ の f を変えると異なる u, v が生成されるため異なる関係式が得られる．この操作を $n+2$ 個の関係式が得られるまで繰り返す．$n+2$ 個の関係式が集まれば平方差法を用いて N の分解を試みる．

3.1.2 $u - vN$ について

Schnorr の素因数分解アルゴリズムで関係式を得るためには $|u-vN|$ が p_n-smooth な数になる必要がある．$|u - vN| \le p_n$ のときは明らかに p_n-smooth である．一方で，$|u - vN| > p_n$ のときに $|u - vN|$ が p_n-smooth になる確率について考える．$u = \log_{p_n}(|u - vN|)$ と置くと $p_n^u = |u - vN|$ となる．ここで $\psi(p_n^u, p_n)$ について考えると式 (2.3) と式 (2.4) より，十分大きな p_n^u に対して以下の式が成り立つ．

$$\frac{\psi(p_n^u, p_n)}{p_n^u} \le \frac{1}{\Gamma(u+1)} \tag{3.2}$$

これより $|u-vN|$ が大きくなるほど u の値が大きくなるため，$|u-vN|$ が p_n-smooth になる確率は小さくなる．よって Schnorr の素因数分解アルゴリズムで関係式を確実に得るためには $|u-vN| \leq p_n$ にする必要がある．式 (3.1) を式変形して $u-vN$ の形で表すと次のようになる．

$$\sum_{i=1}^{n}(z_i C \ln p_i) - C \ln N = \epsilon$$

$$\Rightarrow \frac{u}{vN} = e^{\frac{\epsilon}{C}} \tag{3.3}$$

$$\Rightarrow u = e^{\frac{\epsilon}{C}} vN$$

$$\Rightarrow u - vN = (e^{\frac{\epsilon}{C}} - 1)vN \tag{3.4}$$

式 (3.4) の右辺を p_n 以下の値に近づけることができれば関係式を確実に得ることができるので，ここからはパラメータ n, C を操作することで式 (3.4) の右辺を小さな値にすることができるか調べる．

第4章　最短ベクトルに対するu-vN

4.1　最短ベクトルに対するu-vNの値

ここからはベクトル$\mathbf{zB}$を格子上の最短ベクトルとして考え，このときに$z_1, \ldots, z_n, \epsilon$が満たす条件について述べる．ベクトル$\mathbf{zB}$のノルムは以下のようになる．

$$\|\mathbf{zB}\| = \sqrt{(z_1 f(1))^2 + \cdots + (z_n f(n))^2 + \epsilon^2}$$

このとき，格子$\mathcal{L}(\mathbf{B})$の体積は基底行列$\mathbf{B}$が正方行列の三角行列であることより$\mathrm{vol}(\mathcal{L}(\mathbf{B})) = n!C \ln N$なので，定理2.1.4より以下の式が成り立つ．

$$\|\mathbf{zB}\| < \sqrt{n+1}(n!C \ln N)^{\frac{1}{n+1}}$$

これより次の不等式が成り立つ．

$$z_1, \ldots, z_n, \epsilon < \sqrt{n+1}(n!C \ln N)^{\frac{1}{n+1}} \tag{4.1}$$

4.2　Cのみ増加させた場合

ここではパラメータn, Cを操作した場合に式(3.4)の右辺を小さい値にすることが可能か調べる．まずはnを固定しCのみ増加させた場合を考える．式(4.1)よりϵに関して以下が成り立つ．

$$\epsilon = O(C^{\frac{1}{n+1}}) \tag{4.2}$$

vは式(4.1)より次のように表せる．

$$\begin{aligned} v &= \prod_{z_i<0} p_i^{-z_i} \\ &= \prod_{z_i<0} e^{-z_i \ln p_i} \\ &= O(e^{C^{\frac{1}{n+1}}}) \end{aligned}$$

これより式(3.4)の右辺は次のように表せる．

$$(e^{\frac{\epsilon}{C}} - 1)vN = O((e^{C^{-\frac{n}{n+1}}} - 1)e^{C^{\frac{1}{n+1}}})$$

ここで関数 $f(x)$ を次のように定める．

$$f(x) = (e^{x^{-\frac{n}{n+1}}} - 1)e^{x^{\frac{1}{n+1}}}$$

このとき以下の定理が成り立つ．

定理 4.2.1.
関数 $f(x)$ を

$$f(x) = (e^{x^{-\frac{n}{n+1}}} - 1)e^{x^{\frac{1}{n+1}}}$$

とする．このとき以下の式が成り立つ．

$$\lim_{x\to\infty} f(x) = \infty$$

Proof.
ロピタルの定理を用いるため $g(x)$ を次のようにおく．

$$f(x) = \frac{g(x)}{h(x)}$$
$$g(x) = (e^{x^{-\frac{n}{n+1}}} - 1)$$
$$h(x) = e^{-x^{\frac{1}{n+1}}}$$

このとき以下が成り立つ．

$$\lim_{x\to\infty} g(x) = \lim_{x\to\infty} h(x) = 0$$

$g(x)$ と $h(x)$ を微分すると以下のようになる．

$$g'(x) = -\frac{n}{n+1}x^{-\frac{2n+1}{n+1}}e^{x^{-\frac{n}{n+1}}}$$
$$h'(x) = -\frac{1}{n+1}x^{-\frac{n}{n+1}}e^{-x^{\frac{1}{n+1}}}$$

これより以下の式が成り立つ．

$$\lim_{x\to\infty} \frac{g'(x)}{h'(x)} = \frac{ne^{x^{-\frac{n}{n+1}}+x^{\frac{1}{n+1}}}}{x} = \infty$$

よってロピタルの定理より以下が成り立つ．

$$\begin{aligned}\lim_{x\to\infty} f(x) &= \lim_{x\to\infty} \frac{g(x)}{h(x)} \\ &= \lim_{x\to\infty} \frac{g'(x)}{h'(x)} \\ &= \infty\end{aligned}$$

□

定理 4.2.1 より，n を固定して C のみ増加させた場合 $|u - vN|$ は小さな値に収束しないことが分かる．

4.3 n と C を変化させた場合

次に n と C を変化させた場合に式 (3.4) の右辺が収束するか調べる．式 (4.1) より ϵ と v は次のように表せる．

$$\epsilon = O(\sqrt{n+1}(n!C)^{\frac{1}{n+1}}),\ v = O(e^{\sqrt{n+1}(n!C)^{\frac{1}{n+1}}})$$

よって式 (3.4) の右辺は次のように表せる．

$$(e^{\frac{\epsilon}{C}} - 1)vN = O((e^{\sqrt{n+1}(n!C^{-n})^{\frac{1}{n+1}}} - 1)e^{\sqrt{n+1}(n!C)^{\frac{1}{n+1}}}) \tag{4.3}$$

ここで以下の関数 $f(x, y)$ を考える．

$$f(x, y) = (e^{\sqrt{x+1}(x!y^{-x})^{\frac{1}{x+1}}} - 1)e^{\sqrt{x+1}(x!y)^{\frac{1}{x+1}}} \tag{4.4}$$

$y = x$ としたとき式 (4.4) は次のように表せる．

$$f(x) = (e^{\sqrt{x+1}(x!x^{-x})^{\frac{1}{x+1}}} - 1)e^{\sqrt{x+1}(x!x)^{\frac{1}{x+1}}}$$

このとき以下の定理が成り立つ．

定理 4.3.1.
関数 $f(x)$ を

$$f(x) = \left(e^{\sqrt{x+1}(x!x^{-x})^{\frac{1}{x+1}}} - 1\right)e^{\sqrt{x+1}(x!x)^{\frac{1}{x+1}}}$$

とする．このとき以下の式が成り立つ．

$$\lim_{x\to\infty} f(x) = \infty$$

Proof.
次の関数 $g(x)$ を考える．

$$g(x) = \sqrt{x+1}\left(x!x^{-x}\right)^{\frac{1}{x+1}}$$

このとき定理 2.3.3 より以下のように表せる．

$$\begin{aligned} g(x) &= \sqrt{x+1}\left(x!x^{-x}\right)^{\frac{1}{x+1}} \\ &\sim \sqrt{x+1}\left(\left(\frac{x}{e}\right)^x x^{-x}\right)^{\frac{1}{x+1}} \\ &= \sqrt{x+1}\left(\frac{1}{e}\right)^{\frac{x}{x+1}} \end{aligned}$$

両辺対数とると以下のようになる.

$$\ln g(x) \ \sim \ \frac{1}{2}\ln(x+1) - \frac{x}{x+1}$$

上式の右辺は $x \to \infty$ のとき ∞ に発散するため以下の式が成り立つ.

$$\lim_{x\to\infty} \ln g(x) = \infty \Rightarrow \lim_{x\to\infty} g(x) = \infty$$

関数 $g(x)$ が発散するということは関数 $f(x)$ のかっこの中は発散するため以下の式が成り立つ.

$$\lim_{x\to\infty} f(x) = \infty$$

□

$y = x$ のとき式 (4.4) の関数 $f(x,y)$ は発散することが分かったので $f(x,y)$ の式の形から $y < x$ で増加させたときも発散することが分かる.

次に式 (4.4) に対して $y = e^{x+1}$ とすると次のように表せる.

$$f(x) = \left(e^{\sqrt{x+1}x!^{\frac{1}{x+1}}e^{-x}} - 1\right) e^{\sqrt{x+1}(x!)^{\frac{1}{x+1}}e}$$

このとき以下の定理が成り立つ.

定理 4.3.2.

関数 $f(x)$ を

$$f(x) = \left(e^{\sqrt{x+1}x!^{\frac{1}{x+1}}e^{-x}} - 1\right) e^{\sqrt{x+1}(x!)^{\frac{1}{x+1}}e}$$

とする. このとき以下の式が成り立つ.

$$\lim_{x\to\infty} f(x) = \infty$$

Proof.

まず関数 $f(x)$ は定理 2.3.3 を用いて以下のように式変形できる.

$$\begin{aligned}
f(x) &= \left(e^{\sqrt{x+1}x!^{\frac{1}{x+1}}e^{-x}} - 1\right) e^{\sqrt{x+1}(x!)^{\frac{1}{x+1}}e} \\
&\sim \left(e^{\sqrt{x+1}\left(\frac{x}{e}\right)^{\frac{x}{x+1}}e^{-x}} - 1\right) e^{\sqrt{x+1}\left(\frac{x}{e}\right)^{\frac{x}{x+1}}e} \\
&> \left(e^{x^{\frac{3x+1}{2(x+1)}}e^{-\frac{x(x+2)}{x+1}}} - 1\right) e^{x^{\frac{3x+1}{2(x+1)}}e^{\frac{1}{x+1}}} \\
&> \left(e^{xe^{-\frac{x(x+2)}{x+1}}} - 1\right) e^{xe^{\frac{1}{x+1}}} \\
&> \left(e^{xe^{-(x+2)}} - 1\right) e^{xe^{\frac{1}{x+1}}}
\end{aligned}$$

ここで，関数 $g(x), h(x), k(x)$ を以下のように置く．

$$g(x) = (e^{xe^{-(x+2)}} - 1)e^{xe^{\frac{1}{x+1}}}$$
$$h(x) = e^{xe^{-(x+2)}} - 1$$
$$k(x) = e^{-xe^{\frac{1}{x+1}}}$$

このとき以下の式が成り立つ．

$$\lim_{x\to\infty} h(x) = \lim_{x\to\infty} k(x) = 0$$

また $h(x), k(x)$ の微分は以下のようになる．

$$h'(x) = \left(\frac{1}{x} - 1\right) xe^{xe^{-(x+2)} - (x+2)}$$
$$k'(x) = \left(\frac{1}{(x+1)^2} - \frac{1}{x}\right) xe^{-xe^{\frac{1}{x+1}} + \frac{1}{x+1}}$$

よって

$$\frac{h'(x)}{k'(x)} = \frac{\left(\frac{1}{x} - 1\right)}{\left(\frac{1}{(x+1)^2} - \frac{1}{x}\right)} e^{x(e^{-(x+2)} + e^{\frac{1}{x+1}} - 1)} e^{\frac{1}{x+1}} e^{-2}$$

ここで次の関数 $S(x)$ を考える．

$$S(x) = x(e^{-(x+2)} + e^{\frac{1}{x+1}} - 1)$$

そして関数 $S_1(x), S_2(x)$ を次のようにおく．

$$S_1(x) = e^{-(x+2)} + e^{\frac{1}{x+1}} - 1$$
$$S_2(x) = x^{-1}$$

このとき以下の式が成り立つ．

$$\lim_{x\to\infty} S_1(x) = \lim_{x\to\infty} S_2(x) = 0$$

また $S_1(x), S_2(x)$ を微分すると次のようになる．

$$S_1'(x) = -e^{-(x+2)} - \frac{1}{(x+1)^2} e^{\frac{1}{x+1}}$$
$$S_2'(x) = -x^{-2}$$

ここで $S_1'(x)/S_2'(x)$ を考えると以下が成り立つ．

$$\frac{S_1'(x)}{S_2'(x)} = \frac{x^2}{e^{x+2}} + \frac{x^2}{(x+1)^2} e^{\frac{1}{x+1}}$$
$$\Rightarrow \lim_{x\to\infty} \frac{S_1'(x)}{S_2'(x)} = 1$$

よってロピタルの定理より

$$\lim_{x\to\infty} S(x) = \lim_{x\to\infty} \frac{S_1'(x)}{S_2'(x)} = 1$$

これより $h'(x)/k'(x)$ は次のようになる．

$$\frac{h'(x)}{k'(x)} = \frac{\left(\frac{1}{x}-1\right)}{\left(\frac{1}{(x+1)^2}-\frac{1}{x}\right)} e^{S(x)} e^{\frac{1}{x+1}} e^{-2}$$

$$\Rightarrow \lim_{x\to\infty} \frac{h'(x)}{k'(x)} = \infty$$

よってロピタルの定理より

$$\lim_{x\to\infty} g(x) = \lim_{x\to\infty} \frac{h'(x)}{k'(x)} = \infty$$

$f(x) > g(x)$ より

$$\lim_{x\to\infty} f(x) = \infty$$

□

定理 4.3.1 と定理 4.3.2 より，C より n を速く増加させた場合と C を n よりも速く指数的に増加させた場合に対して $u - vN$ は収束しないことが分かった．この結果からパラメータ n, C を操作することにより $|u - vN|$ を p_n 以下の値にすることはできないため，Schnorr の素因数分解アルゴリズムでは確実に関係式を得ることは難しいと考えられる．

第5章　実験結果

ここでは実際にSchnorrの素因数分解アルゴリズムを実装し素因数分解実験を行った結果について述べる．アルゴリズムの実装はSageMath[7]で行い，使用した計算機のCPUはAMD EPYC 7H12 CPU @ 2.6GHzである．アルゴリズム内部で格子上の短いベクトルを得る際にはSageMathで実装されているBKZアルゴリズム[8]を用いた．パラメータは$C = N$とし，nとNのビット数を同じ値に設定している．実験では，格子を生成し短いベクトルを得て関係式が得られるか判定するところまでの流れを1ループとし，必要な個数の関係式が集まるまでのループ数を測定する．必要な数の関係式が集まらず素因数分解できなかった場合は，最後に関係式が得られたときのループ数を記録する．合成数のビット数は20ビットから80ビットまで10ビット間隔で設定している．実験を行った結果を表5.1に示す．

表 5.1: 素因数分解実験結果

n	$N = p \times q$	ループ数	得た式の数	結果
20	997517 $= 977 \times 1021$	255	22	○
30	926619433 $= 30047 \times 30839$	1789	32	○
40	612145380227 $= 928651 \times 659177$	26342	42	○
50	607866613162429 $= 23248319 \times 26146691$	427949	52	○
60	596119758828291667	3564162	15	×
70	7774854035228626488 11	1850167	2	×
80	9137986412933197167 40109	0	0	×

実験では最大で50ビットの合成数の素因数分解に成功した．そして必要な個数の関係式が集まるまでのループ数は合成数のビット数とnが増加するとともに大きくなるという結果を得た．60ビット以上の合成数については計算機の使用時間制限により必要な個数の関係式が集められていない．60ビットと70ビットの合成数については関係式がいくつか得られているため時間をかければ素因数分解ができる可能性があると考える．80ビットの合成数については関係式が得られていないため時間をかけたとしても素因数分解ができる可能性があるかどうか不明であ

る．この実験結果から Schnorr の素因数分解アルゴリズムで数百ビットの大きな合成数に対して関係式を集めることは難しいと考えられる．

第6章　おわりに

Schnorr の素因数分解アルゴリズムは $|u - vN|$ が p_n-smooth な数であれば関係式が得られ素因数分解を行うことができるため，パラメータ n, C を操作した時に $|u - vN|$ を小さな値に収束させることができるかどうか検証を行った．その結果 n を固定し C のみを増加させた場合，$|u - vN|$ は発散することがわかった．また，C よりも n を速く増加させた場合と C を n の指数オーダーの速度で増加させた場合のどちらを考えたときも $u - vN$ は発散した．ただし，この結果は最短ベクトルの上界から v, ϵ のオーダーを考えたときの結果である．v, ϵ がそれより小さいのであれば結果は異なってくる可能性がある．しかし検証結果から少なくとも n と C を増加させても $|u - vN|$ が小さい値に収束する根拠は現在のところないことはわかる．この結果，$\log_{p_n}(|u - vN|)$ の値も小さな値に収束しないため $|u - vN|$ が p_n-smooth になる確率は低いといえる．したがって Schnorr の素因数分解アルゴリズムで関係式を確実に得られるとはいえない．また実際にアルゴリズムの実装を行い素因数分解実験を行ったところ 50 ビットまでの合成数しか素因数分解できなかったため，数百ビットの大きな合成数の素因数分解は Schnorr の素因数分解アルゴリズムでは難しいと考えられる．

謝辞

指導教員の藤崎英一郎先生には研究や日々の生活において数えきれないほど多くの助言をいただき本当にお世話になりました.

藤崎研究室の学生の皆様にはゼミ等での議論や日頃の何気ない雑談を通して研究活動を行うための大きな活力をいただきました.

お世話になった皆様方に心より深く感謝申し上げます.

参考文献

[1] P. Zimmermann. *[Cado-nfs-discuss] Factorization of RSA-250*, (2020). `https://sympa.inria.fr/sympa/arc/cado-nfs/2020-02/msg00001.html`.

[2] A.K. Lenstra, H.W. Lenstra, M.S. Manasse, and J.M. Pollard. The number field sieve. *STOC 1990*, pp. 564–572, (1990).

[3] C.P. Schnorr. Fast Factoring Integers by SVP Algorithms, corrected. *Cryptology ePrint Archive*, (2021).

[4] C.P. Schnorr. Factoring Integers by CVP Algorithms. *Number Theory and Cryptography*, Vol. 8260, pp. 73–93, (2013).

[5] A. Granville. Smooth numbers: computational number theory and beyond. *Algorithmic Number Theory*, Vol. 44, pp. 267–323, (2008).

[6] A. Hildebrand and G. Tenenbaum. Integers without large prime factors. *Journal de théorie des nombres de Bordeaux*, Vol. 5, No. 2, pp. 411–484, (1993).

[7] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.0)*, (2023). `https://www.sagemath.org`.

[8] C.P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, Vol. 66, pp. 181–199, (1994).