

FUN-ECMプロジェクト

FUN-ECM Project

中島俊平
Nakajima Shunpei

福永慧
Fukunaga Kei

小澤貴也
Ozawa Takaya

落合航平
Ochiai Kouhei

金子真澄
Kaneko Masumi

水上敬介
Mizukami Keisuke

外山拓
Toyama Taku

1. 概要

1.1 プロジェクト名について About project name

FUN-ECMのFUNは、はこだて未来大学を、ECMは Elliptic Curve Method(楕円曲線法)を意味しており、それらを繋げたのが本プロジェクト名の由来である。

Why this project name is FUN-ECM is that FUN means Future University Hakodate and ECM means Elliptic Curve Method that is a best algorithm for prime factorization.

1.2 背景 Background

約40年前に RSA暗号※1が登場し、安全な電子商取引が可能となった。RSA暗号の安全性は公開鍵の素因数分解の困難さ等価であると考えられるため、RSA暗号の普及以降、素因数分解実験は重要なテーマとなっている。本プロジェクトが素因数分解法の中でECMを対象とする理由は、最良法の1つであることや並列実装のしやすさ、近年普及している楕円曲線暗号(ECC)※2が関連しているためである。

※1 素因数分解が困難であればあるほど安全である公開鍵暗号の一つ。

※2 楕円曲線上の離散対数問題の困難性を安全性の根拠とした公開鍵暗号の一つ。ECMと楕円曲線暗号のアルゴリズムは類似している。

RSA encryption ※1 was developed at 40 years ago, and secure e-commerce has become possible. Prime factorization experiments have become an important theme since the spread of RSA. Because the security of RSA cryptosystem is considered to be equivalent difficulty in prime factorization of public key". We have three reasons that prime factorization among ECM is selected. First, it is one of the best methods. Second, it is easy to implement in parallel. Third, elliptic curve cryptography (ECC) ※2 have become widespread relate in recent years.

※1 RSA is one of public key cryptographies. RSA is based on the hardness of prime factorizations.

※2 ECC is one of public key cryptographies. ECC is based on the hardness of elliptic curve discrete logarithm problems. Algorithm for ECM is similar to one of ECC.

1.3 目的 The purpose

- | | |
|-----------------------------|--|
| ・ECMを利用した素因数分解のプログラムの作成 | ・Making of program for prime factorization using ECM. |
| ・素因数分解チャレンジECMNETにランクイン(※1) | ・Ranking in ECMNET, which is a factorization challenge |
| ・STUDIO KAMADA(※2)にランクイン | ・Ranking in STUDIO KAMADA. |
| ・活動や結果を発信・報告するサイトを作成 | ・Making a website to report activities and results |

※1:ECMNETの目的は、主にECMにより大きな合成数の因数を見つけることで

Cunningham projectに貢献すること。(詳細はQRコードを参照)

The purpose of ECMNET is to find large factors by ECM, mainly by contributing to the Cunningham project.

URL: <https://members.loria.fr/PZimmermann/records/ecmnet.html>

※2:STUDIO KAMADAの目的は、ニアレプディジット(ゼロ目の自然数)関連の数の素因数分解表を作成する。(詳細はQRコードを参照)

The purpose of STUDIO KAMADA is to create a prime factorization table of numbers related to near replicates (natural number of Repdigit).

URL: <http://stdkmd.com/nrr/#summary>



ECMNET



STUDIO KAMADA

1.4 今期活動成果

STUDIO KAMADAに載っているまだ素因数が
見つかっていなかった合成数2つの素因数分解に成功した。
詳細は、QRコードを参照。

52×10²⁴⁶⁺¹¹ 9 の素因数分解

- 素因数分解 - STUDIO KAMADA into English



目次

- 1 現状の状況
- 2 2 の素因数分解を解する手法
- 3 ECM による試み
- 4 素因数分解の完成の経緯
- 5 予言する

1. 現状の状況

$$\frac{52 \times 10^{246+11}}{9} = 5(71419_{247}, 7) \times 7 \times 103 \times 117720023928963_{110} \times 2673346968987650140392955711549594667693170506907_{-612} \times$$

$$\frac{125664794510901221755377(11)5465240274532864482304818768027515240251324431187333998102443688812508771520826328907913877598044049118713557954403532346515274369186822442646_{-1793} | \text{Gnucfn2017} / \text{ECM R1} \times 432000000 \text{ for } p52 / 2017 \text{ 年 } 11 \text{ 月 } 11 \text{ 日}$$

[Go to Table](#)

$76 \times 10^{247-31}$
9

の素因数分解 - 素因数分解 - STUDIO KAMADA into English

目次

1. 現在の状態

1. 現在の状態



$76 \times 10^{247-31}$
9

= 8(4)2461+246t = 3 × 85711 × 650112876289+123t × 1337489853071+123t × 91585637851471680049375897339286952181012351+...+t

41238857794821264032084280293156521930414208610581736039625265489895869897562117786406120359399247094739424483056287740347725271466624745292305302432756815902519255676657133+... (Runcom2017 / ECM B1-43000000 for P44 × P175 / 2017 年 11 月 1 日)

2. 活動内容

2.1 年間スケジュール	前期活動内容			夏季休業		後期活動内容		
	5月	6月	7月	8月	9月	10月	11月	12月
プログラム班	基礎学習	BSGSを実装	中間発表 報告書準備	計測		Montgomery 曲線の実装	最終発表 準備	最終報告
理論班		BSGS理解				素数ベアリングの 実装 FFTの理解	最終発表 準備	

FUN-ECMプロジェクト

FUN-ECM Project

中島俊平

Nakajima Shunpei

福永慧

Fukunaga Kei

小澤貴也

Ozawa Takaya

落合航平

Ochiai Kouhei

金子真澄

Kaneko Masumi

水上敬介

Mizukami Keisuke

外山拓

Toyama Taku

2.2 後期活動内容詳細

プログラムの計算時間を短縮させるために、様々なアルゴリズムの実装を行った。他にも、シェルスクリプトやドキュメントの作成した。

・素数テーブル

ECMではやや小さな素数のテーブル(例えば43億以下の素数のテーブル)が必要となる。去年までのプログラムでは素数を求めるのにC言語のライブラリを使用していたが、それでは素数を求めるのに時間がかかってしまうため、予めテキストファイルに素数を用意して、素数が必要になった際はそのファイルを読み込み、計算に用いる。

・新たなアルゴリズムの実装

プログラムの計算の速度を向上させるために、様々なアルゴリズムを実装した。アルゴリズムはBaby-Step Giant-Step、モンゴメリー曲線、素数ペアリングを使用した。

・シェルスクリプトの作成

プログラムの実行を効率良く行うために、スクリプトを作成した。実行された結果をテキストに出力するようにしており、指定した桁数の合成数を計算できるようにした。

3. ECMについて

3.1 基礎学習

目的

昨年度のプログラムを理解するために、メンバー全員が、基礎知識として楕円曲線を知る必要があった。

方法

『楕円曲線暗号入門(2013年度)』(著 伊豆哲也)を用いて学習した。

学習内容

・ Z/nZ の世界での加算、減算、乗算、除算について

$Z/nZ = \{0, 1, 2, \dots, n-1\}$ までの整数の集合を考えると、加算・減算・乗算は普通に計算した結果の $\text{mod } n$ をとる。除算について、 $a, b \in Z/nZ$ に対して、 $a \div b$ の解は $a = bc \text{ mod } n$ となる c である。 n が素数の場合、 $b \neq 0$ ならば $a \div b$ の計算が可能

・楕円曲線の定義(方程式)

楕円曲線 E は、方程式 $E: y^2 = x^3 + ax + b$ で定義される。

・楕円曲線 E 上の2点 P, Q の加法

・ $P + Q$ の定義: P, Q を通る直線 l と楕円曲線の交点を R とすると、点 R と x 軸に関して対称な点 R' のこと。
・ $2P$ の定義: 直線 l を点 P における接線とする。さらに、この接線と楕円曲線の交点を R とすると、点 R と x 軸に関して対称な点 R' のこと。 P の座標と Q の座標から $P + Q$ の座標や $2P$ の座標を計算する公式が存在する(加算公式と2倍算公式)。

・スカラー倍

・ E の点 P と自然数 n から、 n 個の P の和 $nP = P + P + \dots + P$ をスカラー倍という。
・ECMの支配的な演算はスカラー倍である。本プロジェクトにとって、スカラー倍の高速化は重要である。高速化の方法として、バイナリ法や射影座標の手法がある。

・射影座標

点 (x, y) を z 座標を追加して (x, y, z) で表す座標を射影座標といい、射影座標を用いるとスカラー倍が高速になる。

・Stage1 とは

素因数分解したい合成数 N に対して、適切な大きさの B_1 から k を求め、 $\text{mod } N$ 上でスカラー kP を計算する。そして、 kP の z 座標と N との最大公約数が1でなければ N の因数である。見つからなかったらStage2へ。
見つかる理由として、 P を N の素因数の1つとすると、 m が $\#E \text{ mod } p$ の倍数の時、 $mP = O \text{ (mod } p)$ となり、 $mp = (x: y: z) \text{ mod } N$ に対して、 $z \equiv 0 \text{ (mod } p)$ となる。

・Stage2 とは

$B_1 < s < B_2$ を満たす、すべての素数 $s_1, s_2, s_3, \dots, s_k$ までのStage1で得られた点を Q として全てのスカラー倍 $s_i Q$ を求め、 $s_i Q$ の z 座標と N との最大公約数をとる。Stage2を高速に行う方法にBaby-Step Giant-Step法がある。

3.2 ECMの計算の流れ

Stage1の流れ

$P \in E \text{ mod } N$ に対して、 $Q = mP = (x, y, z) \text{ mod } N$ を計算する

$\text{gcd}(n, z) \neq 1$ の時
終了

$\text{gcd}(n, z) = 1$ の時
 Q を返して、stage2へ

Stage2の流れ

Stage1で見つかった Q をもとに、Baby-Step Giant-Step法を使用(Baby-Step Giant-Step法については3.3.2を参照)

未発見の時、曲線を変えてStage1へ移行

発見した時、終了

FUN-ECMプロジェクト

FUN-ECM Project

中島俊平

Nakajima Shunpei

福永慧

Fukunaga Kei

小澤貴也

Ozawa Takaya

落合航平

Ochiai Kouhei

金子真澄

Kaneko Masumi

水上敬介

Mizukami Keisuke

外山拓

Toyama Taku

3.3 アルゴリズム

3.3.1 Montgomery曲線について

Montgomery曲線とは、 $By^2 = x^3 + Ax^2 + x$ という式から表される曲線である。メリットとして、 y 座標を保存する必要がなくなるため、Weierstrass曲線と違って、 y 座標を用いずに計算を出来るようになる。デメリットとして今まで使用した曲線と違って $P - Q$ の点を用意する必要がある。

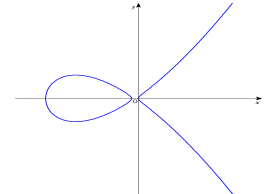
Montgomery曲線の座標の計算について

・2点同士の加算

点 P と Q を加算するためには以下のアルゴリズムに従って加算を行う。

点 $P = (Xp:Zp)$, 点 $Q = (Xq:Zq)$ とする。 $P_0 = (Xp - q:Zp - q) = P - Q$ とする。

これらを用いて計算を行う。



Montgomery曲線

・Montgomery ladder

Montgomery ladderはMontgomery曲線のスカラー倍を効率よく行うためのアルゴリズムである。

3.3.2 Baby-Step Giant-Step法(以下,BSGS)について

去年までは、Stage2の部分では効率の悪い総当たり法が実装されていた。

そこで今年は、より速く計算できるBSGSを実装した。

BSGSとは？

$B_1 < s < B_2$ を満たす全ての素数を $s_1, s_2, s_3, \dots, s_k$, Stage1の結果で得られた点 P として、

s を変形して計算を行うことで高速化を図る手法である。

Stage2は $s * Q = O$ のとき成功する。この式を変形していく。 $s = 210 * v + u$ ($-105 < u < 105$) とすると

$(210 * v) * P = -u * P$ となる。

この式を満たせばよいので、楕円曲線の逆元の定義より、 $\text{mod } p$ 上で、右辺と左辺の x 座標が等しい。

$(210 * v) * P$ の x 座標を Gx 、 $-u * P$ の x 座標を Hx とすると

$Gx \equiv Hx \pmod{p}$ となり、これは $Gx - Hx \equiv 0 \pmod{p}$ を意味し、

$(Gx - Hx)$ が p の倍数となる。ここで、全ての s について $(Gx - Hx)$ を計算し、

すべてを掛け合わせた数を d をすると、 $Gx - Hx \equiv 0 \pmod{p}$ となる s が存在すれば、 $d \equiv 0 \pmod{p}$ となる。

そのため、 d と合成数 N の最大公約数を求めることで p が判明する。

3.3.3 素数ペアリングについて

BSGSをより高速化する手法として、素数ペアリングがある。

$(210 * v) * P = -u * P \cdots \star$

☆式の計算をについて考える。

楕円曲線の逆元の定義より、 $-u * P$ と $u * P$ は、 x 軸に対して対称であり、 x 座標の値が等しい。よって、 $(Gx - Hx)$ の値は同じになる。

もし、 $210 * v + u$, $210 * v - u$ がともに素数だったとき、従来の計算手法だと、同じ計算を二重に行うことになり効率が悪い。

そこで、 $210 * v + u$ と $210 * v - u$ がともに素数になるペアを計算し、 s がどちらかの時のみ $(Gx - Hx)$ を求める。

これにより、試行回数が減るため高速化することができる。

☆式はこのようにも変形可能である。

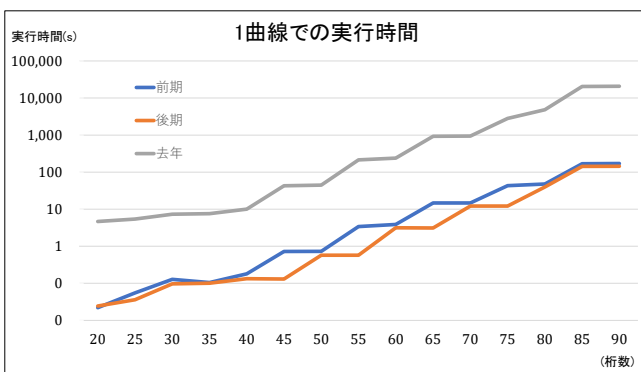
もし、 $s = 210 * v + u$ は素数だったが、 $210 * v - u$ が素数でないためにペアリングできなかったときに、

$s = 210 * v + u = 210 * (v - 1) + (210 + u) = 210 * (v - 2) + (2 * 210 + u) = 210 * (v - 3) + (3 * 210 + u) = \dots$

と考えることで、より深くペアを探すことができ、ペアの数が増え、試行回数がより減るため、BSGSの高速化が期待できる。

ただし、☆式の右辺が増える分、多くの事前計算が必要となるため、注意が必要になる。

4. 計測結果



去年と今年の前期と後期に作成したそれぞれのプログラムの1つの楕円曲線にかかる計算時間の平均を比較した。

1つの楕円曲線にかかる計算は今年度作成したプログラムが去年のプログラムより約80倍速度が向上した。

今後の展望として、符号付き2進展開法を実装し、新たなアルゴリズムを実装して素因数を見つける速度を高めていきたい。

ECMNETにランクインすることを試みる。その為、少なくとも桁数128桁以上の合成数の素因数分解をする必要がある。

(2017/12/6 現在)