多倍長平方数の高速探査法

後　保範（神奈川大学）

整数多項式の多倍長平方数を高速に探す方法を見つけた。これを FSS（高速平方数探査法)と名付ける。拡張 Fermat 法($y=(x+M)^2-4abN$, $y>0$)で平方数 y を探す実験をした。2048 ビットの N で平方根を計算して判定する方法に比較し数千万倍高速化した。これより、FSS は RSA 暗号解読において、ふるい法に代わり得る。一方、RSA 暗号解読へ応用する研究はまだ初期段階である。その方法も簡単に紹介する。

Fast search method for multi-precision square number.

Yasunori Ushiro (Kanagawa University)

I found a method to search the square number with multiple precision at high speed on integer polynomial. I call this FSS (Fast Squared Search method). I evaluated to find the square number in the extended Fermat method. Compared with the FSS and calculating the square root with 2048 bits of N, it has accelerated tens of millions of times. Therefore, the FSS considers a method that can replace the sieving method in RSA cryptanalysis. On the other hand, research to apply to RSA cryptanalysis is still in the early stages. The method is briefly introduced.

1. はじめに

ふるい法は整数多項式 g(x)から素数 P が因数となるものを選ぶのに、除算でなく P 飛びに記憶し、それを利用し高速化する。現在の RSA 暗号解読は、この高速性を利用した GNFS (一般数体ふるい法)が使用されている。RSA 暗号解読は合成数 N を $S^2=T^2 \pmod N$となる整数 S,T を探すことに帰着する。そのため、整数多項式の多倍長平方数を高速に探す方法を検討した。平方剰余で平方数になりえないものを除ける。多倍長関数(gmp)に既に値の判定で使用されている。ここでは、平方剰余を値そのものではなく、整数多項式に適用することを考え、FSS(高速平方数探査法)と名付けた。拡張 Fermat 法で、2048 ビットの N の平方根を求めて平方数を判定する場合の百万倍高速化を予測した。最終的に予測を超えた高速化が得られた。そのために多数のアイデアと試作を繰り返したが、その部分はページの都合で述べない。RSA 暗号解読へ応用する研究はまだ初期段階であるが、その一方法も紹介する。

2. 平方数探査の目的

多倍長合成数 N を P と Q に因数分解するには、$S^2=T^2 \pmod N$となる整数 S,T を求める。S 側は平方数やその積で選べるが、T 側を平方数やその積にするのが難しい。MPQS (複数多

項式 2 次ふるい法)では S は平方数の積で、T は因数の各素数の積のベキが偶数になるようにする。GNFS は複雑な手順だが、最終的に $S^2=T^2 \pmod N$にする。FSS を RSA 暗号解読へ応用する一方法を示す。N に対し整数 d を MPQS を応用し、$((A^2x+b)^2-N)/A^2=a^2d$ から複数計算する。d は N の平方根以下の値とする。ここで、$b^2=N \pmod{A^2}$である。整数 d で x を動かし $y=((x+c)^2-N)/d$ を計算し、平方数 y を 2 つ見つければ、N が因数分解される。ここで、$c^2=N \pmod d$であり。Fermat 法と異なり、多数の d で見つけられる。d の選び方がよければ、効率よく N は因数分解される。整数 m は d の大きさで決められる。まだ、効率の良い d を選ぶ方法は研究中である。他にも平方数探査で N を因数分解する方法が多数考えられる。

3. 多項式での平方数と剰余の関係

整数 y が $u^2=y \pmod P$の整数解 u を持つとき 1、そうでないとき 0 とする。y が平方数なら各 P で 1 である。逆に一つの P で 0 なら y は平方数でない。複数の P を使えば、平方数の確率が増す。最初に、各 P で G_Pなる判定テーブルを作成する。次に、多項式 f(x)は、$f(x+P)=f(x) \pmod P$が成り立つので、各 P で F_Pが作成できる。F_Pで平方数候補を絞る。P を奇素数に選ぶと、1 個の P で平方数候補はほぼ半減する。30 個の P なら、候補はほぼ 1/10 億に絞れる。P が合成数やベキ数ならより効率が上がる。表 1 に P=8,15,7,11,13,17 の G_Pと F_Pを示す。$f(x)=(M+x)^2-N$ の例で示す。N=6404633577312547963,M=2530737754 とする。共に P 単位で繰り返し使える。

表 1. 平方数判定テーブル G_P,F_P の例

P	G_P	F_P
8	1 1 0 0 1 0 0 0	1 0 0 0 1 0 0 0
15	1 1 0 0 1 0 1 0 0 1 1 0 0 0 0	0 0 0 1 1 0 0 0 0 1 0 0 0 1 0
7	1 1 1 0 1 0 0	1 0 1 0 1 1 0
11	1 1 0 1 1 1 0 0 0 1 0	0 1 0 1 1 0 1 1 0 1 0
13	1 1 0 1 1 0 0 0 0 1 1 0 1	0 1 0 0 1 0 1 1 1 1 0 1 0
17	1 1 1 0 1 0 0 0 1 1 0 0 0 1 0 1 1	1 1 1 0 0 0 0 0 1 1 1 0 1 0 0 1 0

最初に、P=8 で f(x)が平方数候補を拾うと、x=0,4 となる。これは 8 で繰り返す。次に P=15 で評価すると、x=4,24,28,48,64,84,88,108 が得られ、120 で繰り返す。x<240 で評価すると、P=7 で x=4,28,84,124,144,168,184,208,228 と 9 個得られる。次いで、P=11,13 及び 17 で評価すると一つの x=144 に絞られる。これより、$(M+x)^2-N=731060910441=855021^2$ となる。S=2530737898. T=855021 とすると、$S^2=T^2 \pmod N$が得られる。

4. 拡張 Fermat 法の平方数探査

合成数 N の 2 つの素因数が極端に近いとき Fermat 法で因数分解できる。2 つの素因数の比が整数 a と b の比に極端に近いとき使用できる方法を拡張 Fermat 法と呼ぶ。この方法は、$(M+x)^2-4abN$ の平方数を探す。M は 4abN の平方根を切り上げた整数とする。拡張 Fermat 法

は N の 8 乗根近い比の整数 a,b が分からないと RSA 暗号解読には使用できない。一方、平方数となるものを多数の中から探す、平方数探査の比較に適している。今回、FSS の高速性の評価は拡張 Fermat 法で行った。拡張 Fermat 法は Fermat 法を含む。平方数と剰余の関係では $f(x)=(M+x)^2-N$ を、$f(x)=(M+x)^2-4abN$ と置き換えると、そのまま適用できる。G_P は各 P で固定であるが、F_P は f(x)に依存して変わる。

5. RSA 暗号解読への応用例

RSA暗号解読への応用研究は始めたばかりであり、ふるい法と同じように発展することを期待している。そのために、一方法を記載する。下記の MPQS(複数多項式ふるい法)で整数 x を動かし、分解対象数 N の平方根より少し小さい d を複数求める。A を A^2 にするのが特徴。

$((A^2x+b)^2-N)/A^2=a^2d, \quad b^2=N \pmod{A^2}$ --- (1)

各 d に対し、整数 x を動かし $y=((dx+c)^2-N)/d$ を計算する。ここで、$c^2=N \pmod{|d|}$である。y が平方数になるものを2つ見つければNは因数分解できる。dは合成数のためcは多数となる。平方数の一方は(4-1)式に一致し、新しい関係は一つである。2 つの c,x を c_1,x_1 と c_2,x_2 で表し、y の平方根を h_1,h_2 とすると下記が成立する。

$(dx_1+c_1)^2=dh_1^2 \pmod N, \quad (dx_2+c_2)^2=dh_2^2 \pmod N$ --- (2)

(2)式から $S^2=T^2 \pmod N$が得られる。ここで、$S=(dx_1+c_1)h_2$ で $T=(dx_2+c_2)h_1$ である。以下同じ N を 2 ケースの A で分解する例を示す。N=229910794091155831 とする。

＜ケース 1＞　A は素数 2731 に選ぶ。

A^2=7458361, b=1774180 となり、x=51 で a=15,d=-49979238 から(1)式が成り立つ。

c は 8 個で、c_1= 6444595,c_2= 32295925 から x_1=1,x_2=7 で h_1= 67353,h_2=40965 となる。これから、$25738988755623^2=2311402318845^2 \pmod N$となり、N=455570569×504665599 と分解される。$A^2x+b=|dx_2+c_2|$= 382150591 及び $Aa=h_2$=40965 から二つ目が(1)式に一致する。

＜ケース 2＞　A は素数 2861 に選ぶ。

A^2= 8185321, b= 1209257 となり、x=-78 で a=9,d= 265714130 から(1)式が成り立つ。

c は 16 個で、$c_1=c_2$= 105817521 から x_1=2,x_2=22 で h_1= 25749,h_2= 363921 となる。この場合もケース 1 と同じく、N=455570569×504665599 と分解される。一つ目が(1)式に一致する。

6. 数値実験結果

数値実験は下記の環境で行った。

PC: HP Desktop 870 (Intel Core i7 6700k), 4Gh, 8GB, システムは Windows10

コンパイラー: Cygwin Ver. 2.7 gcc(64bit 版), 最適化(-O3), 多倍長計算: gmp の mpz_t

現在の RSA 暗号と同じ 2048 ビットの N を使用し、Fermat 法と拡張 Fermat 法で測定した。拡張 Fermat 法は a=1017,b=1231 で $y=(M+x)^2-4abN$ である。測定は共に 3 方式で行った。表中の平方根は平方数の判定に平方根計算をした。剰余法は gmp の剰余を利用して平方数を判定する関数を使用した。FSS は各 N で因数分解の時間を測定した。他の 2 方法は百億回の

探査時間を測定した。表 2 に Fermat 法の結果を、表 3 に拡張 Fermat 法の結果を示す。FSS は評価する関数の計算が数百回で済むため、N のビット数に計算時間は依存しない。一方、N(Fermat 法)又は 4abN(拡張 Fermat 法)が 2,3,5,7 及びそのベキ数で平方剰余かどうかが FSS の探査速度に影響する。a,b が 2,3,5,7 を因数に含まなければ、拡張 Fermat 法の方が、Fermat 法より 1 秒間の探査回数は多い傾向がある。平方根と剰余法は N の性質には依存せず、ビット数が倍になると、3～4 倍遅くなる。平方根及び剰余法はデータに依存しないので、A-1 と B-1 だけ示す。データは番号が 1 増えると、4 倍探査回数が増えるものを使用。

表 2. Fermat 法による平方数探査速度(N は 2048 ビット)

方式	データ	探査数 (百億個)	計算時間 (s)	1 秒間計算 (億回)	高速化比(倍)	
					平方根	剰余法
平方根	A-1	1	5,147	0.02	1	---
剰余法	A-1	1	1,335	0.07	4	1
FSS	A-1	900,000	135	700,000	34,000,000	9,000,000
	A-2	3,600,000	594	600,000	31,000,000	8,000,000
	A-3	14,400,000	2,499	600,000	30,000,000	8,000,000
	A-4	57,600,000	7,497	800,000	40,000,000	10,000,000

表 3. 拡張 Fermat 法による平方数探査速度(N は 2048 ビット、a=1017,b=1231)

方式	データ	探査数 (百億個)	計算時間 (s)	1 秒間計算 (億回)	高速化比(倍)	
					平方根	剰余法
平方根	B-1	1	5,300	0.02	1	---
剰余法	B-1	1	1,465	0.07	4	1
FSS	B-1	1,800,000	112	1,600,000	85,000,000	23,000,000
	B-2	7,200,000	561	1,300,000	68,000,000	19,000,000
	B-3	28,700,000	2,377	1,200,000	64,000,000	18,000,000
	B-4	114,700,000	5,656	2,000,000	108,000,000	30,000,000

7. 終わりに

拡張 Fermat 法(含む Fermat 法)で FSS の高速性を評価した。予測以上の効果で、2048 ビットの N に対し、平方根を計算する方式の 3 千万倍～1 億倍高速になる。多倍長計算ライブラリ(gmp)は剰余を利用して平方数を判定する関数がある。それと比較しても 8 百万倍から 3 千万倍高速になる。FSS は平方数判定を PC で 1 秒間に 60 兆回～200 兆回することができる。

FSS は分解対象 N のビット数に計算時間は依存しない。N の性質(2,3,5,7 とそのベキ数の平方剰余)で数倍の差がでる。今回の結果から、FSS は RSA 暗号解読で現在のふるい法に代わる可能性があると思われる。一方、RSA 暗号解読へ応用する研究はまだ初期段階である。その一つのアイデアを記載した。π 計算世界記録は AGM(算術幾何平均化法)から DRM(分割有理数化法)に移った。同様に、RSA 暗号解読の記録を GNFS(一般数体ふるい法)から FSS を利用した方式に移すのが夢である。