素因数分解計算機実験
研究調査報告書

２００４年1月２３日

立教大学

木田 祐司

素因数分解計算機実験

研究調査報告書

2004年1月23日
2004年3月3日修正

立教大学 木田祐司

1　前書き

RSA 暗号、特に 1024-bit RSA、の安全性についてはいくつかの予測が発表されている。中には 2001 年までしか安全性が確保できないというような報告 [1] もあったが 2004 年 1 月現在ではまだまだ安全である。このプロジェクトはその予測の根拠となるデータを実際にプログラムを作り、計算機を動かして収集することを主たる目的とする。

また収集したデータから 1024-bit RSA について考察を試みた。しかしながら遥か先の 1024-bit RSA について予測を行うことは多くの不確定要素のためにほとんど不可能である。我々は実験データをプロットし、理論的に予想される式を当てはめて単純にグラフを延長してみた。どこまでが実際的でどこからが空想的であるかは我々が決めるところではない。最終的な判断は他の文献、たとえばLenstra 達のレポート [7] などと比較して読者自らが行っていただきたい。

2　問題の設定

RSA 暗号の安全性は素因数分解の困難性に基づいている。そこで以下、この報告では次のような数 n が与えられたときに、その素因数 p, q を知ること（これが n の素因数分解）の困難性について調べる。

合成数 n は同程度の大きさの二つの異なる素数 p, q の積である。

このような数の素因数分解の難易度を評価する方法には大きく分けて三種類ある。

(a) 現在存在しているハードウェア上でプログラムを作成して実験を行う。

(b) 現在の技術で作成可能だが実際には作られていないハードウェアを用いて思考実験を行う。

(c) 現在の技術では実現不可能なハードウェアを用いて思考実験を行う。

(b) に属するものとしては [2] や [3] の組み合わせなどがある。実現可能性を含めた議論は別のプロジェクトで行われており、ここでは扱わない。(c) の代表は量子計算機であるがこれも別のプロジェクトにお任せしたい。したがって我々が対象とするのは (a) である。

(a) の方法は昔からいろいろと考案されてきたが、上のような数を素因数分解するには「一般数体ふるい法(General Number Field Sieve method)」をおいて他には考えられない。以下ではこれを GNFS と略記する。

3　これまでの評価

1024 ビット合成数の素因数分解計算量については、先に述べたようにRSA 暗号や署名で用いられるため、もし簡単にできるものだとすると社会的影響も大きいことから、これまでにも多くの人が評価を行なってきた。2002 年度末までの評価については [4, Section 2.4.1] に詳しくまとめられているので、本稿ではそれ以降の結果についてまとめる。

4　プロジェクト遂行の手順

このプロジェクトの目的を達するには世界水準のプログラムを作って実験をしなければいけない。理論的な準備と付随的なプログラムの作成は昨年度までで終えていたので今年度は主たるプログラムの開発を行った。

1. 多項式選択
2. lattice sieve
3. filtering
4. block Lanczos
5. 代数的数の平方根計算

の5つである。

出来上がったプログラムを用いて我々は世界第二位の素因数分解記録を達成した。この結果は我々のプログラムが世界水準に達していることを証明しており、我々の行った実験の速度が現時点ではベストに近いものであることを保証している。この分解の詳細については最後の節で報告する。

大きな数を分解したことによって実験の対象となる数に対する分解は見通しの良いものとなり、多くの実験をスムーズに行うことができた。

5 GNFS の概略

GNFS の詳細についてはたとえば我々が以前に提出した報告書

`http://www.shiba.tao.go.jp/kenkyu/CRYPTREC/PDF/outrep_data/rep_ID0021.pdf`

を参照することにしてここでは後の議論に必要な部分だけを解説する。

まず、分解対象となる n が与えられたとき、多項式 $f(X)$ と整数 M を

$$f(M) \equiv 0 \bmod n$$

となるように選ぶ。この多項式の選び方により全体の計算時間が左右される。

$f(X) = 0$ の解となる複素数 θ をとり、代数体 $K = \mathbf{Q}(\theta)$ を作る。K の整数環を O_K とする。

適当な個数の有理整数の集合(rational factor base) RFB と適当な個数の O_K の1次素イデアルの集合(algebraic factor base) AFB をとる。

a, b を互いに素な有理整数で $a + bM$ は RFB-smooth とする。つまり

$$a + bM = \prod_{p \in RFB} p^{e(p)}$$

であり、かつ $a + b\theta$ は AFB-smooth とする。つまりこれが生成するイデアルは

$$(a + b\theta)O_K = \prod_{\mathcal{P} \in AFB} \mathcal{P}^{e(\mathcal{P})}$$

と完全に素イデアル分解されるものとする。

このようなペア (a, b) を $\#RFB + \#AFB$ 個より多く集める。

> このとき、 smooth であるものだけを集めるために「エラトステネスのふるい」に類似した手法を用いる。それが「数体ふるい法」という名前の由来である。以下ではこの (a, b) を集める段階のことを単に「ふるい」と記す。

すると各べき指数を mod 2 で考えたベクトル

$$\left((e(p) \bmod 2)_{p \in RFB},\ (e(\mathcal{P}) \bmod 2)_{\mathcal{P} \in AFB} \right)$$

は線形従属となり、適当に組み合せると、次のような関係式が出る。

このような組み合せを見つけるには行列を作って解を求めることになる。しかしそのまま行列を作ると大きくなりすぎるので予備的な処理を施す。この作業を以下では「filtering」と記す。また filtering で小さくなった行列を普通はblock Lanczos 法で解く。以下ではこの段階のことを「行列計算」と記す。

$$\prod(a_i + b_i M) = \left(\prod_{p \in RFB} p^{e(p)} \right)^2$$

と $\mathbf{Z}$ で平方数になり、かつイデアルの積

$$\prod(a_i + b_i \theta) O_K = \left(\prod_{\mathcal{P} \in AFB} \mathcal{P}^{e(\mathcal{P})} \right)^2$$

はと K で平方イデアルになる。こういう関係式をいくつか集めて平方イデアルと平方数で生成されるイデアルの違いを補正すれば平方数を作ることができる。

$$\prod(a_i + b_i \theta) = h(\theta)^2$$

であったとしてこれを準同型

$$\begin{array}{rcl} \phi : \mathbf{Z}[\theta] & \longrightarrow & \mathbf{Z}/N\mathbf{Z} \\ g(\theta) & \mapsto & g(M) \bmod n \end{array}$$

で写せば

$$\left(\prod_{p \in RFB} p^{e(p)} \right)^2 \equiv h(M)^2 \bmod n$$

となる。これから両辺の平方根の差と n の最大公約数をとれば n の約数を得る。これが 1 または n 自身であったときは別の組み合わせを試すことになる。

ここで a, b を単純に動かすのではなく、ある素イデアル $Q = (q, \theta - r)$ に含まれる数のみを動かす方法を lattice sieve という。この方法の利点はすべての数が Q で割れているのでそれだけsmooth になる確率が高まることである。このときの Q を special-Q と呼ぶ。

6 GNFS の理論的な実行時間

GNFS により n を素因数分解するとき、factor base の個数、「ふるい」の範囲を共に

$$L_n[1/3, (8/9)^{1/3}]$$

とすれば「ふるい」の実行時間はおおよそ

$$L_n[1/3, (64/9)^{1/3}]$$

と見積もられる。ただし記号 $L_x[\ ,\ ]$ の定義は次の通りである。

$$L_x[\nu, \lambda] = \exp\left((\lambda + o(1))(\log x)^{\nu} (\log\log x)^{1-\nu} \right)$$

この評価式は n が大きくなるときの漸近的な評価であって注意深く扱わねばならない。たとえば n が大きくなると、多項式の次数 d もそれに応じていくらでも大きくなるのであるが我々の扱う n については d は 4, 5, 6 と非常に小さい。これではこの評価式をそのまま良い近似値を与える式として使うことはできない。そこで我々はこの評価式の元となった次数別の評価式を採用することにした。

特定の次数を用いたときの GNFS の実行時間は [5, Conjecture 11.4, p.81] で考察されている。それによれば d 次多項式を用いた場合の実行時間は

$$\exp\left((1+o(1))\left(d\log d+\sqrt{(d\log d)^2+4\log(n^{1/d})\log\log(n^{1/d})}\right)\right)$$

である。Murphy [6, Section 3.1] は多項式の最高次係数の大きさが $n^{1/(d+1)}$ くらいとなることから

$$\exp\left((1+o(1))\left(d\log d+\sqrt{(d\log d)^2+4\log(n^{1/(d+1)})\log\log(n^{1/(d+1)})}\right)\right)$$

とするべきであると指摘している。我々が採用するのはこの式である。

7　実験データの概略

我々が行った実験のデータから抜粋して掲載する。詳細は共同研究者からのレポートとして別に提出されることとなっている。

	hc	hd	#q	#relations	ふるい時間	行数	行列時間
gnfs90d4	2 048	1 024	37 000	1 422 612	13 570	119 970	
gnfs90	2 048	1 024	48 000	1 422 465	14 350	105 898	
gnfs90d6	2 048	2 048	92 000	2 551 165	46 621	174 649	
rsa100d4	2 048	2 048	95 000	2 736 929	55 155	215 271	
rsa100f	2 048	2 048	112 000	3 053 728	54 271	190 616	00:07:19(*)
rsa100d6	2 048	2 048	240 000	5 329 204	145 247	336 137	
rsa110d4	4 096	2 048	220 000	6 185 022	250 945	384 185	
rsa110g	2 048	2 048	200 000	6 752 059	193 676	304 917	00:28:49(*)
rsa110d6	4 096	2 048	430 000	10 228 570	461 685	528 397	
rsa120d4	4 096	4 096	580 000	13 145 521	1 320 933	741 068	
rsa120	4 096	2 048	650 000	14 225 875	790 138	687 247	02:24:15
rsa120d6	4 096	4 096	720 000	19 304 631	1 517 888	904 269	
rsa130	4 096	4 096	1 000 000	26 975 303	2 315 347	1 249 886	08:47:28
rsa140g	8 192	8 192	904 000	51 340 137	8 728 209	1 842 573	15:07:39(*)
rsa150	8 192	8 192	2 200 000	124 804 557	20 597 260	4 061 097	101:31:17(*)

注意：行列時間で (*) がついているものは若干異なる行列のものである。

`hc, hd` は「ふるい」領域の縦と横である。`#q` は special-Q の個数である。ふるい時間は「ふるい」にかかった時間を秒単位で表している。また行数は「行列計算」を行った行列の行数であり、行列時間は16台のPCで並列に行った「行列計算」の実行時間を 時:分:秒 で表している。

また最初のラベル名に `d4, d6` がついているものはそれぞれ4次、6次多項式を用いた場合である。それ以外は5次多項式を用いたものであり、`f, g` などがついたものは複数行った実験のひとつを表す。

8 ふるいの時間の近似式の決定

我々の実験で「ふるい」にかかった時間を t（単位 CPU・時）とし、t を理論式

$$C_d \exp\left((1+o_d(1))\left(d\log d+\sqrt{(d\log d)^2+4\log(n^{1/(d+1)})\log\log(n^{1/(d+1)})}\right)\right)$$

で近似するときに適切な値 C_d, $o_d(1)$ を決定する。

実験値をまとめると次のようになっている。単位は CPU・時 である。

nの桁数	4次式	5次式	6次式
90	3.8	4.0	13.0
100	15.3	15.0	40.4
110	69.7	53.8	128.3
120	367.0	219.5	421.6
130		643.1	
140		1,868.4	
150		5,721.5	

これらの値にあてはまる単一の $o_d(1)$ は存在しないがおおよそ -0.10 から -0.06 が近いようである。それぞれに対して C_d を決める。

$o_d(1)=-0.10$ とするとき $C_4=3.0\cdot10^{-12}, C_5=2.0\cdot10^{-12}, C_6=1.2\cdot10^{-12}$ が適当である。このとき計算値は次のようになる。単位は CPU・時 である。

nの桁数	4次式	5次式	6次式
90	3.5	4.0	13.5
100	16.0	15.0	41.9
110	68.5	53.2	125.2
120	277.3	180.9	363.5
130	1,072.2	592.8	1,026.1
140	3,975.5	1,877.0	2,822.7
150	14,191.0	5,761.5	7,581.1

$o_d(1)=-0.06$ とするとき $C_4=9.4\cdot10^{-13}$, $C_5=5.4\cdot10^{-13}$, $C_6=3.1\cdot10^{-13}$ が適当である。このとき

計算値は次のようになる。単位は CPU・時 である。

n の桁数	4 次式	5 次式	6 次式
90	3.8	3.8	13.3
100	18.5	15.1	43.2
110	84.2	56.7	135.8
120	362.8	203.8	413.3
130	1,489.7	703.8	1,221.8
140	5,854.7	2,345.6	3,515.8
150	22,114.7	7,568.1	9,866.5

9 ふるいの時間の予測

この節では前節で得られた近似式を単純に延長してより大きな n の素因数分解の「ふるい」の実行時間について考察する。

重要な注意
この節での予測は明示的、暗示的を問わず多くの現実的、非現実的な仮定の下での結論である。
現時点における実現の可能性について著者は関知しない。

前章で得られた近似式をそのまま先へ延長してみる。
$o_d(1) = -0.10$ の場合は次のようになる。単位は CPU・年 である。

n の bit 数	4 次式	5 次式	6 次式
512	3	1	1
576	29	8	8
640	270	61	48
704	2,281	405	260
768	17,768	2,515	1,333
832	128,895	14,698	6,488
896	877,446	81,312	30,161
960	5,640,142	428,108	134,444
1024	34,412,866	2,154,445	576,613

となる。
$o_d(1) = -0.06$ の場合は次のようになる。単位は CPU・年 である。

n の bit 数	4 次式	5 次式	6 次式
512	5	1	2
576	52	12	12
640	529	98	75
704	4,907	708	436
768	41,867	4,767	2,403
832	331,676	30,130	12,553
896	2,458,786	179,857	62,481
960	17,167,392	1,019,495	297,642
1024	113,512,363	5,512,610	1,361,886

となる。

このように 512-bit 以上では 4 次式はまったく不利であり、576-bit までは 5 次式が勝り、そこからは 6 次式が勝ると推測される。したがって 1024-bit の合成数を GNFS で素因数分解するには 6 次式を使うのが良く、そのとき「ふるい」を 1 台の PC で行うと 50 万年から 140 万年かかると推測される。また「ふるい」は独立した計算に分割できるので同時に稼動させる PC の台数に比例して計算の量が増える。そのため 50 万台から 140 万台の PC を同時に稼動させれば 1 年で「ふるい」を終える計算になる。

ただし後述するように 300 ギガバイトの主記憶が必要となる。また 1024-bit では factor base の素数の大きさも 32-bit を越えて 37-bit ほどになるのでその理由でも `seti@home` のような Internet による分散計算に直ちに乗せることはできない。

後述の Lenstra 達のレポート [7] では 1024-bit の「ふるい」時間を我々のおよそ 200 倍から 600 倍多く見積もっている。

10 行列の大きさの近似式の決定

我々の実験で用いた行列の行数を r（単位 1000）とし、r を理論式

$$C_d \exp\left((0.5 + o_d(1))\left(d\log d + \sqrt{(d\log d)^2 + 4\log(n^{1/(d+1)})\log\log(n^{1/(d+1)})}\right)\right)$$

で近似するときに適切な値 C_d, $o_d(1)$ を決定する。

実験値をまとめると次のようになっている。単位は 1000 である。

桁数	4 次式	5 次式	6 次式
90	120	106	175
100	215	190	336
110	384	304	528
120	741	687	904
130		1,250	
140		1,842	
150		4,061	

これらの値にあてはまる単一の $o_d(1)$ は存在しないがおおよそ -0.06 から -0.04 が近いようである。それぞれに対して C_d を決める。

$o_d(1) = -0.06$ とするとき $C_4 = 1.2 \cdot 10^{-4}$, $C_5 = 1.0 \cdot 10^{-4}$, $C_6 = 7.3 \cdot 10^{-5}$ が適当である。このとき計算値は次のようになる。単位は 1000 である。

桁数	4 次式	5 次式	6 次式
90	96	104	176
100	200	197	305
110	407	366	521
120	807	666	877
130	1,563	1,189	1,457
140	2,967	2,089	2,389
150	5,527	3,614	3,873

$o_d(1) = -0.04$ とするとき $C_4 = 6.1 \cdot 10^{-5}$, $C_5 = 5.1 \cdot 10^{-5}$, $C_6 = 3.6 \cdot 10^{-5}$ が適当である。このとき計算値は次のようになる。単位は 1000 である。

桁数	4次式	5次式	6次式
90	90	99	169
100	195	194	301
110	410	371	526
120	838	693	908
130	1,673	1,271	1,542
140	3,270	2,292	2,587
150	6,265	4,065	4,287

11 行列のサイズの予測

この節では前節で得られた近似式を単純に延長してより大きな n の素因数分解での行列の行数について考察する。

重要な注意
この節での予測は明示的、暗示的を問わず多くの現実的、非現実的な仮定の下での結論である。
現時点における実現の可能性について著者は関知しない。

前節で得られた近似式をそのまま先へ延長してみる。
$o_d(1) = -0.06$ の場合は次のようになる。単位は 10^6 である。

n の bit 数	4次式	5次式	6次式
512	7	5	5
576	23	13	12
640	67	33	28
704	191	84	63
768	522	204	140
832	1,375	483	304
896	3,512	1,115	644
960	8,723	2,513	1,337
1024	21,118	5,537	2,724

となる。

$o_d(1) = -0.04$ の場合は次のようになる。単位は 10^6 である。

n の bit 数	4 次式	5 次式	6 次式
512	8	5	5
576	27	15	14
640	86	41	33
704	255	108	79
768	728	276	183
832	2,003	679	410
896	5,340	1,628	899
960	13,820	3,806	1,930
1024	34,831	8,692	4,061

となる。

このように 512-bit 以上では 4 次式はまったく不利であり、576-bit 未満では 5 次式が勝り、そこからは 6 次式が勝ると推測される。したがって 1024-bit の合成数を GNFS で素因数分解するには 6 次式を使うのが良く、そのとき行列の行数は $2.7 \cdot 10^9$ から $4 \cdot 10^9$ 程度になると予想される。

12 行列の計算時間の予測

我々の実験から行列の計算時間を行数の単純な式で表すことは実験例が少ないこともあってうまくいかなかった。しかし n が大きくなるにつれて一行当たりの非ゼロ要素の個数は増大する傾向にあることは間違いないから Lanczos 法の仕組みにより計算時間は行数の 2 乗で下から押さえられる。

後述の 164 桁の分解のときは行数が $7 \cdot 10^6$ で実行時間は約 12 日であった。そこで 1024-bit の場合の行数を $4 \cdot 10^9$ とすると少なくとも

$$\left((4 \cdot 10^9)/(7 \cdot 10^6)\right)^2 \cdot 12 \text{ 日} = 10,735 \text{ 年}$$

はかかる計算になる。ただし、行列の 1 行当たりの非ゼロ要素の個数を 1000 とすれば行列を格納するだけで 16T バイトの主記憶が必要になる。16 台に分散するので 1 台あたりでは 1T バイトである。ワークエリア等を含めるとこの倍の量が必要となる。

しかしこれは 16 台の PC をギガビット・イーサネットで接続したクラスターの場合である。より高速なネットワークに、より多くの PC を接続すればそれだけ短時間で終了し、1 台当たりの主記憶量も少なくすることができるであろう。

上記 12 日のうち 4 日はネットワークの通信時間であったので、1024-bit の場合の通信時間は

$$\left((4 \cdot 10^9)/(7 \cdot 10^6)\right)^2 \cdot 4 \text{ 日} = 3,578 \text{ 年}$$

であるが 10 倍速いネットワークであればこれが 358 年に減ることになる。

次に PC の台数を増やすことを考える。行列の計算は「ふるい」と違って密結合のネットワーク上で行う必要がある[1]。したがって接続可能な台数には制約がある。また台数を $p = q^2$ とすると計算時間は $1/p$ に比例して減るが、通信時間は $1/q$ でしか減らないという原理的に大きな問題もある。

$64^2 = 4,096$ 台は実際に構築できるであろう。このとき計算時間は $7157/(4096/16) = 28.0$ 年に減るが通信時間は $358/\sqrt{4096/16} = 22.4$ 年にしか減らない。合計で 50.4 年となる。

[1]ただし、任意の 2 台が直に接続されている必要はなく、格子状に配列した場合に縦位置が同じ任意の 2 台、横方向が同じ任意の 2 台が直に接続されていれば良い。トーラス状のネットワークというイメージである。

台数を $16q^2$ とすると合計の時間は $7157/q^2 + 358/q$ 年となる。これを 1 年にするには $q = 377.0$、つまり $2,273,881$ 台が必要となる。

まとめると 1024-bit の素因数分解に必要な行列計算は

- 4096 台の PC を 10 ギガビット・イーサネットで接続したクラスタで 50 年以上かかる。
- 227 万台の PC を 10 ギガビット・イーサネットで接続したクラスタで 1 年以上かかる。

と見積もられる。

いずれにしろこれまで使われてきたような通常の型の計算機によっては現実的な時間内に処理できないことも現実的なコストで実現できないことも確実である。この理由で専用ハードウエアによる解決の道が模索されいくつかの有望な提案が行われている。しかしながらこれは我々のプロジェクトの守備範囲外であり、別のプロジェクトの調査研究にお任せすることにする。

13 主記憶の量について

ここでは「ふるい」で使用する主記憶の量について考察する。主記憶は大きくは factor base の記憶用と「ふるい」領域の二つに分かれる。前者は構造体の構成によるが (#RFB+5#AFB) 個の記憶領域が必要である。我々の実験の範囲では 1 個は 4-byte である。後者はさらに「ふるい」領域とワークエリアに分かれるがどちらも factor base の記憶領域と同じくらいになる。ただし「ふるい」領域は全領域を分割することにより数分の 1 で済ませることもできる。

factor base の個数は行列の行数にほとんど一致する。したがって 1024-bit の分解には $4 \cdot 10^9$ 個の素数と素イデアルからなる factor base を用いることになるが、一つの素数は 5-byte の大きさが必要なので全部で $100 \cdot 10^9$-byte が必要になる。「ふるい」領域およびワークエリアと合わせると 300 ギガバイトの主記憶を使うことになる。現在のサーバ・マシンは主記憶が 512 ギガバイトあるのが普通なので多くはあるが不可能な量ではない。

14 最近の他のレポートについて

Lenstra 達によるレポート [7] では 1024-bit 素因数分解における「ふるい」について実際に計算機実験を行なうことにより評価を行なっている。

このレポートでは理論的な漸近式

$$C_R(n) = L_n[1/3, (64/9)^{1/3}]$$

を用いて、実際に分解を実行した n の時間 R_n から別の数 n' にかかる時間 $R(n')$ を $\frac{C_R(n)}{C_R(n)} R_n$ として予測する方法は n と n' が同程度の大きさでない場合は $o(1)$ を考慮しないと誤った結果を導く可能性があると指摘している。このようなことを避けるために Lenstra 達は多くのパラメータを変化させ、場合によっては実際に smoothness を確認することにより 1024-bit 分解の「ふるい」に必要な計算量を出している。

彼らはまず最初に次数 5 から 9 の多項式を生成し、全ての評価の基本としている。次に、Dickman の ρ 関数と、large prime を rational side と algebraic side に 1 つずつ用いた場合について、様々な「ふるい」領域と、factor base について数値計算を行ない、効果が高そうなパラメータを選択している。次に、ここで得られたパラメータの近傍について、より詳細にパラメータを動かし、また実際に smoothness を確認することにより「ふるい」領域から得られる relation 数を、いくつかの代表点を選ぶことにより導出している。この結果によると、ρ を使った評価より概ね少ない relation しか得られていないが、少ないところでも

80%程度で押えられている。結果、多項式の次数については $d = 6$ か 7 辺りが適しているとの結果が出ている。また、具体的に明記されている計算量やパラメータについては明示されていない。

一方、[8] の発表では 512-bit GNFS、1024-bit SNFS、1024-bit GNFS について

$$
\begin{array}{lrcl}
\text{512 GNFS} & & & \\
\downarrow & 6 & \times & \text{factor base size} \\
 & 700 & \times & \text{effort} \\
\text{1024 SNFS} & & & \\
\downarrow & 140 & \times & \text{factor base size} \\
 & \text{5e5} & \times & \text{effort} \\
\text{1024 GNFS} & & &
\end{array}
$$

とのまとめが示されている。しかし、但し書きがついていて

> much smaller effort with larger factor bases

とあることにも注意を払う必要がある。

ここで 512-bit GNFS と 1024-bit GNFS を直接比較すると factor base size で 840 倍、effort（計算時間） で $3.5e8$ 倍ということになる。我々の単純な延長法とくらべると factor base size についてはほぼ一致するが effort については Lenstra 達は我々の単純な延長による推測より 200 ～ 600 倍ほど多くの時間がかかると見積もっている。

15 164 桁の素因数分解について

最初に述べた通り、このプロジェクトの目的を達するには世界水準のプログラムを作って実験をしなければいけない。また実際にそのようなプログラムを作ったことを客観的に証明するには多くの人のチャレンジに耐えてまだ素因数分解できていない数を分解してみせなければならない。そのために我々は有名な Cunningham project の Table の中から選んだ数で世界記録を破ることを目標とした。残念ながら分解が終了する前により大きな記録が作られてしまったので世界新記録の達成は持ち越しとなったが世界で二番目の記録ということは我々のプログラムが世界水準にあることを証明するに十分であると信じている。

以下、その分解についてレポートする。

2003 年 12 月 19 日、我々のチームは $2^{1826} + 1$ の約数である 164 桁の合成数を GNFS で分解した。これは一般数体ふるい法による 2 番目に大きな分解である。

対象となった数は以下のものである。

```
c164 =
97583673891702451651094659376990232450771614233997\
85325879473864808161869141292797373828054892194274\
08614639625580843270216606887108154934492304270050\
90485779545989
```

我々はこれが 68 桁と 97 桁の二つの素数の積であることを明らかにした。小さい方の素因数は次の通りである。

```
p68 =
34334644886182446546273008924242084634327089789559\
771215864092254849
```

多項式は次のものを採用した。これは Montgomery-Murphy の方法を一部簡略した方法を用いて選択した。

```
f(x) =
8293702863045600 x^5
+5627796025215486707 x^4
+5575245564273099319021 11 x^3
+17691721660250881843016 1036 x^2
-13601173202899548432935219131949 x
-12171622290476241497444980012311021

M = 41126877593272575259693918 4846
```

「ふるい」の領域は

$$[-16384, 16384) \times [0, 8192)$$

である。special-Q には $43 \cdot 10^6$ から $194 \cdot 10^6$ までの $8.2 \cdot 10^6$ 個の素イデアルを用いた。

rational factor base は $40 \cdot 10^6$ 未満の素数からなり、algebraic factor base は special-Q 未満の素数からなる。

Large prime の限界は両方ともに $4 \cdot 10^9$ である。

「ふるい」にはおよそ一ヶ月を要した。これは Pentium-4 2.53GHz に1G RAM を搭載した PC 一台に換算すると 7 年かかることになる。

Lattice sieve は 409,711,156 個の relation を見つけ、line sieve は 48,732,057 個の relation を見つけた。

Filtering はこれらを 7,501,898 行 7,500,801 列の行列に集積した。この非ゼロ要素の個数は 1,251,138,505 である。

Block Lanczos 法はこの行列（の定める連立方程式）を約 12 日で解いた。これにはギガビットイーサネットで接続された 16 台の PC（「ふるい」で用いたのと同等のもの）を用いた。Lanczos 法の block size は 128 とした。

平方根の計算には Montgomery-Nguyen 法を用いた。

16 結論

本報告で、1024-bit の素因数分解の計算量を見積もった。現在、市販されている汎用のハードウェア (パソコンなど) を用い、2 年程度で分解を完了するためには、一企業や一研究グループでは集められないような数の計算機、場所、電力、ネットワークその他の資源を要することが推測される。しかし、これらのハードウェアの進歩は非常に速いので今後の発展状況を注意深く見積もる必要がある。

参考文献

[1] Arjen K. Lenstra and Eric R. Verheul, *Selecting Cryptographic Key Sizes*, Journal of Cryptology, 14(4)2001, 255–293.

[2] Adi Shamir and Eran Tromer, *Factoring Large Numbers with the TWIRL Device*, Advances in Cryptology — CRYPTO 2003, LNCS 2729(2003), 1–26, Springer-Verlag.

[3] Daniel J. Bernstein, *Circuits for integer factorizaion: a proposal*, 2002. available at `http://cr.yp.to/factorization.html#nfscircuit`,

[4] CRYPTREC Report 2002, Information-technology Promotion Agency, Japan and Telecommunications Advancement Organization of Japan, 2003, Japanese version is available at `http://www.shiba.tao.go.jp/kenkyu/CRYPTREC/fy15/cryptrec20030512_report044.htm`

[5] Arjen K. Lenstra and Hendrik W. Lenstra, Jr., *The development of the number field sieve*, Lecture Notes in Mmath., 1554(1993), Springer-Verlag.

[6] Brian Antony Murphy, *Polynomial Selection for the Number Field Sieve Integer Factorisation Algorithm*, PhD Thesis, 1999, Australian National University, `http://web.comlab.ox.ac.uk/oucl/work/richard.brent/ftp/Murphy-thesis.ps.gz`

[7] Arjen K. Lenstra and Eran Tromer and Adi Shamir and Wil Kortsmit and Bruce Dodson and James Hughes and Paul Leyland *Factoring Estimates for 1024-Bit RSA Modulus*, Advances in Cryptology — ASIACRYPT 2003, LNCS 2894(2003), 55–74, Springer-Verlag

[8] Arjen K. Lenstra, *SNFS versus GNFS and the feasibility of factoring a* 1024*-bit number with SNFS*, 2003, available at `http://homepages.cwi.nl/~herman/Lenstra.ppt`