社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－ＣＳＥＣ－38 (40)
2007／7／20

平方数を探索する素因数分解アルゴリズム

小林 邦勝†

† 山形大学工学部 〒992-8510 山形県米沢市城南 4-3-16
E-mail: †kobayash@yz.yamagata-u.ac.jp

あらまし 合成数 n に関する平方数を探索する素因数分解アルゴリズムを提案する. n の 10 進桁数を $\log_{10} n$ で表し, メモリー量 (計算機の台数) を $(\log_{10} n)^s$, 計算機 1 台当たりの計算量を $(\log_{10} n)^t$ とすると, 素因数分解に要する関係式として,$s+t=1/2$ が得られる. 例えば,$s=1/6$ のとき $t=1/3$ となり, 数体ふるい法とほぼ同じ計算量になる. また,$s=1/4$ のとき $t=1/4$ となり, 並列計算を行うことにより高速化をはかることができる.
キーワード 素因数分解アルゴリズム, 平方数, 並列計算, 拡張 Fermat 法, 相加平均, 相乗平均

A Factoring Algorithm Searching Squares

Kunikatsu KOBAYASHI†

† Faculty of Engineering, Yamagata University 4-3-16 Jonan, Yonezawa-shi, 992-8510 Japan
E-mail: †kobayash@yz.yamagata-u.ac.jp

Abstract We propose a factoring algorihm searching squares. By using parallel computing with this factoring algorithm, we can make a speeding up of factoring.
Key words factoring algorithm, square, parallel computing, extended Fermat method, arithmetic mean, harmonic mean

1. まえがき

素因数分解アルゴリズムの多くは, 合成数 n に関する 2 次合同式 $k^2 \equiv l^2 \pmod n$ を何らかの方法で作り, これから $(k \pm l, n)$ を計算して n の素因数を求めている. この平方数の関係式 $k^2 \equiv l^2 \pmod n$ を作る方法は, Factor Base を用いる方法と用いない方法に大別でき,Factor Base を用いる方法の代表的なものに数体ふるい法 [2] や 2 次ふるい法 [1] があり, 用いないものに Fermat 法がある. これらの関係を図 1 に示す。

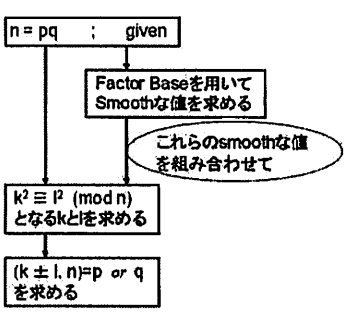

図 1 2 次合同式を用いる素因数分解法

2 次ふるい法は, 有理数の Factor Base, つまり,n に関して平方剰余になる小さな素数の集合を用いて k と l をこれら素数の積で表す方法である. 一方, 数体ふるい法は有理数体の Factor Base を用いて k を作り, 代数体の Factor Base, 例えば,2 次体のノルムが素数となる 2 次整数の集合を用いて l を作る. つまり,2 つの Factor Base を用いて, 両方の要素で k と l を表す方法である.

本文では、Factor Base を用いずに, つまり,Factor Base の要素のみで表される smooth な値を用いないで, 直接 n に関する平方数を探索する素因数分解アルゴリズムを示す. 本方法は, 並列計算に適しており,n の 10 進桁数を $\log_{10} n$ で表すとき, 計算機の台数 (メモリー量) を $(\log_{10} n)^s$,1 台当りの計算機の計算量 (計算時間) を $(\log_{10} n)^t$ とすると,$s+t=1/2$ の関係が得られる. 例えば,$s=1/6$ のとき $t=1/3$ となり, この場合の計算機 1 台当りの計算量は数体ふるい法の計算量にほぼ等しくなる. また,$s=1/4$ のときは $t=1/4$, つまり $(\log_{10} n)^{1/4}$ オーダーとなり, 並列計算に用いる計算台数を増やすことにより, 素因数分解の高速化をはかることができる.

2. 拡張 Fermat 法

合成数 $n=pq$ の素因数 p と q に関しては, 一般に,

$$\text{相加平均}\frac{p+q}{2} \geqq \text{相乗平均}\sqrt{pq} \tag{1}$$

が成立する. 従って, ある正整数 x が存在して

$$x+\lfloor\sqrt{pq}\rfloor=\frac{p+q}{2} \tag{2}$$

となり, このとき,Fermat 法や 2 次ふるい法で用いられる関数 $F(x)$ は

$$\begin{aligned} F(X) &= (x+\lfloor\sqrt{n}\rfloor)^2-n \\ &= (\frac{p+q}{2})^2-pq=(\frac{p-q}{2})^2 \end{aligned} \tag{3}$$

となり, 平方数となる. 従って,2 次合同式

$$(x+\lfloor\sqrt{n}\rfloor)^2=(\frac{p+q}{2})^2\equiv(\frac{p-q}{2})^2 \pmod n \tag{4}$$

が得られ, これから

$$(\frac{p+q}{2}\pm\frac{p-q}{2},n)=p \ \ or \ \ q \tag{5}$$

と n の素因数を求めることができる.

この Fermat 法は次のように拡張される. i と j を正整数とするとき, 式 (1) に対応する次の関係式

$$\text{相加平均}\frac{ip+jq}{2} \geqq \text{相乗平均}\sqrt{ijpq} \tag{6}$$

が得られ,

$$ip\approx jq, \ \ ij\equiv n \pmod 4 \tag{7}$$

である $x=ij$ において

$$\frac{ip+jq}{2}\approx\sqrt{ijpq}=\sqrt{nx}\approx 1+\lfloor\sqrt{nx}\rfloor \tag{8}$$

となる. 従って, 拡張 Fermat 法で用いる関数 $f(x)$ は

$$\begin{aligned} f(x) &= (1+\lfloor\sqrt{nx}\rfloor)^2-nx \\ &= (\frac{ip+jq}{2})^2-ijpq=(\frac{ip-jq}{2})^2 \end{aligned} \tag{9}$$

となり, これから

$$(\frac{ip+jq}{2}\pm\frac{ip-jq}{2},n)=p \ \ or \ \ q \tag{10}$$

と n の素因数を求めることができる. 式 (9) の関係を満たす変数 $x=ij$ は複数存在し, この i と j の比は q と p の比にほぼ等しくなる.

3. 関数 $f(x)=(1+\lfloor\sqrt{nx}\rfloor)^2-nx$ の特性

$n\equiv 1 \pmod 4$ のときの式 (9) の関数 $f(x)$ の概形を図 2 に, $n\equiv 3 \pmod 4$ の場合の概形を図 3 に示す. これらの図からも分かるように, 関数 $f(x)$ は複数の放物線で表される [3]. これらの放物線, すなわち関数 $f(x)$ が極小値をとる変数 x の値は

$$x=\lfloor(\frac{\sqrt{n}-\sqrt{ij}}{2})^2\rfloor, \ \ ij\equiv n \pmod 4 \tag{11}$$

であり,$ip\approx jq$ となる変数 x

$$x=\frac{p-j}{2}\cdot\frac{q-i}{2} \tag{12}$$

において関数 $f(x)$ は平方数となる.

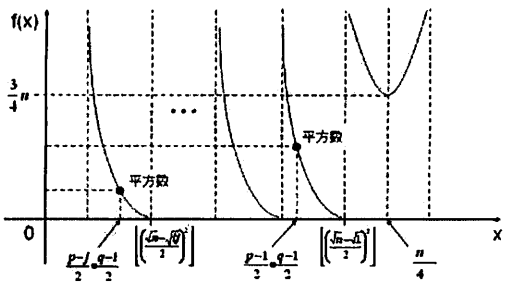

図 2 関数 $f(x)$ の概形 $n\equiv 1 \pmod 4$ の場合

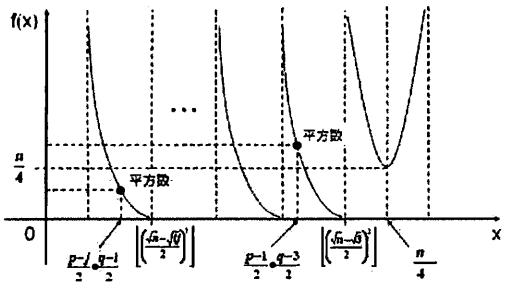

図 3 関数 $f(x)$ の概形 $n\equiv 3 \pmod 4$ の場合

4. 平方数探索アルゴリズム

以下では,$p<q<2p$ の場合に限定して扱う. 従って,$ip\approx jq$ の関係より, $j<i<2j$ となる.

step1. 並列計算に用いる計算機の台数を $(\log_{10}n)^s(o<s<1)$ とするとき, パラメータ j の値をこの $(\log_{10}n)^s$ と同じ桁数 u をもつ適当な整数に定める.

step2. 並列計算を行う各計算機に,$ij\equiv n \pmod 4 (i>j)$ を満たす異なる i の値を与える.(j の値はすべての計算機で同じであり, i の値は計算機毎に異なる.)$n\equiv 1 \pmod 4$ の場合の i の範囲は $j+4\leqq i\leqq 2j-3$ であり, 一方, $n\equiv 3 \pmod 4$ の場合の i の範囲は $j+2\leqq i\leqq 2j-1$ である.

step3. 関数 $f(x)=(1+\lfloor\sqrt{nx}\rfloor)^2-nx$ が極小値となる, 変数 $x=\lfloor(\frac{\sqrt{n}-\sqrt{ij}}{2})^2\rfloor$ における $f(x)$ の値を求める.

step4. この $f(x)$ の値が平方数 l^2 のときは, $(1+\lfloor\sqrt{nx}\rfloor\pm l,n)$ を計算して n の素因数を求める.

step5. $f(x)$ の値が平方数でない場合には,$x=x-1$ として **step3** に戻る.

step6. $x=\lfloor(\frac{\sqrt{n}-\sqrt{ij}}{2})^2\rfloor$ から $x=\lfloor(\frac{\sqrt{n}-\sqrt{ij}}{2})^2\rfloor-j^2$ の範囲で順番に $f(x)$ の計算を行い, この範囲で平方数が得られない場合には, 失敗としてアルゴリズムを終了する.

5. 数 値 例

$p = 549797184491917, q = 834427406578561, n = pq = 458765838799784939134129991437$, $q/p = 1.517700400\cdots$ の場合

step1. n は 10 進 30 桁の数であり, $s = 1/6$ とすると, パラメータ j は 10 進 5 桁の整数となる. ここでは, $j = 54979$ に定める.

step2. $n \equiv 1 \pmod 4$ より, $ij \equiv 1 \pmod 4$, $j < i < 2j$ を満たす i は, $i = 54983 \sim 109955$ の範囲の 4 おきの整数となる. 最初の計算機に $i = 54983$, 2 番目の計算機に $i = 54987$ を与え, 以下, 同様に, $i = i + 4$ を次の計算機の i の値とする. この例では, $i = 83443$ が与えられた計算機について扱う. このときの i と j の比は $i/j = 1.517724950\cdots$ である.

step3. 関数 $f(x)$ が極小値をとる変数 x の値は

$$x = \lfloor (\frac{\sqrt{n} - \sqrt{ij}}{2})^2 \rfloor = 1146914596770080570724691924 87$$

であるが, この x における $f(x)$ の値は平方数とはならない.

step5. $x = x - 1$ として, 関数 $f(x)$ の値を順番に計算すると,

$$x = \frac{p-j}{2} \cdot \frac{q-i}{2} = 274898592218469 \times 417213703247559 = 114691459677008057071718967171$$

のとき, $f(x) = (1 + \lfloor \sqrt{nx} \rfloor)^2 - nx = 185519819081253^2 = l^2$ と平方数が得られる. このときの $1 + \lfloor \sqrt{nx} \rfloor = k$ の値は $k = 22938291937695429185410456185 6$ である.

step4. $(k - l, n) = 549797184491917 = p$, $(k + l, n) = 834427406578561 = q$ と p, q が得られる.

この場合の計算量は, $\lfloor (\frac{\sqrt{n} - \sqrt{ij}}{2})^2 \rfloor - \frac{p-j}{2} \cdot \frac{q-i}{2} = 750225316$ であり, おおよそ 10 進 10 桁 (n の桁数の 1/3) で表される.

次に, 同じ n に対して並列計算機の台数を増やした場合について示す.

step1. $j = 2748989$ とする.

step2. パラメータ i は $i = 274893 \sim 5497977$ の範囲の 4 おきの整数となる. ここでは, $i = 4172141$ とする. このときの i と j の比は $i/j = 1.517700144\cdots$ となり, 小数点以下 6 桁目まで q/p の値に一致する.

step3. 関数 $f(x)$ が極小値をとる変数 x の値は

$$x = \lfloor (\frac{\sqrt{n} - \sqrt{ij}}{2})^2 \rfloor = 1146914585530304533792582776 86$$

であるが, この x における $f(x)$ の値は平方数ではない.

step5. $x = x - 1$ として, $f(x)$ を順番に計算すると

$$x = \frac{p-j}{2} \cdot \frac{q-i}{2} = 114691458553030453379254199440$$

のとき,

$$f(x) = (1 + \lfloor \sqrt{nx} \rfloor)^2 - nx = 96719925185133^2 = l^2$$

と平方数が得られる. このときの $1 + \lfloor \sqrt{nx} \rfloor = k$ の値は $k = 22938291825297668529549426843 7$ である.

step4. $(k - l, n) = 834427406578561 = q$, $(k + l, n) = 549797184491917 = p$ と p, q が得られる.

この場合の計算量は, $\lfloor (\frac{\sqrt{n} - \sqrt{ij}}{2})^2 \rfloor - \frac{p-j}{2} \cdot \frac{q-i}{2} = 4078246$ であり, 10 進 7 桁 (n の桁数の約 1/4.3) となる.

この例からもわかるように, 並列計算機の台数を増やすことにより, パラメータ i と j の比を素因数 q と p の比に近づけることができ, i と j の 10 進桁数が u, p と q の 10 進桁数が v の場合, 素因数を求めるのに要する計算量はおおよそ 10 進 $(v - u)$ 桁で表される.

6. む す び

合成数 n に関する平方数を探索する素因数分解アルゴリズムを提案した. 例えば, 10 進 300 桁の合成数 n の素因数分解を行う場合には, 10^{50} 台の計算機を用いて並列計算を行うと, $O(10^{100})$ (これは n の 10 進桁数の 1/3 乗オーダー) の計算量で n の素因数を求めることができる. 更に, 計算機を増やして, 10^{75} 台の計算機を用いた場合には, $O(10^{75})$ (n の 10 進桁数の 1/4 乗オーダー) の計算量で素因数分解を行うことができる. 一般に, 並列計算機の台数 $(\log_{10} n)^s$ と一台当りの計算量 $(\log_{10} n)^t$ の間には, $s + t = 1/2$ の関係が成立する.

文 献

[1] C.Pomerance, "The Quadratic Sieve Algorithm", Proc. of Eurocrypt '84, LMCS, Vol.209, pp.169-182, (1985).

[2] Lenstra, Lenstra, Manasse, Pollard, "The number field sieve", Proc. 22nd STOC, pp564-572, (1990).

[3] 小林邦勝, "2 次ふるい法を用いた素因数分解アルゴリズム", 信学論 (A)J79-A, 3, pp827-829, (1996).

[4] 小林邦勝, "Fermat 法を拡張した素因数分解アルゴリズム", SCIS 2007, 3A2-1, (2007).