アルゴリズム 46－3
（１９９５．７．２０）

合成数の素因数存在領域を求める 並列処理アルゴリズム

永瀬　宏[†]，武田　直樹[‡]，永井　彰[†]
金沢工業大学　人間・情報・経営系[†]
株式会社シャープ[‡]

整数の素因数分解は、幾何学的にx-y解平面上に双曲線$xy=N$と整数座標の格子点との交点を求めることと解釈できる。本稿では、双曲線$xy=N$付近に存在する整数格子点を求める方法を提案する。この方法では、素因数は最大$N^{1/4}$の間隔で探索することが可能である。最後に、剰余類を考慮した素因数分解の並列処理の可能性を示す。

A Parallel Algorithm to Determine Solution Domain for Prime Factorization of Integers

Hiroshi NAGASE[†], Naoki TAKEDA, and Akira NAGAI[†]
Department of Information and Computer Engineering
Kanazawa Institute of Technology[†],
SHARP Corporation[‡]

Prime Factorization of integers can be explained geometrically as finding, in an x-y solution plane, the crosspoint of hyperbola $xy = N$ and integer coodinate lattice points. The present paper proposes methods to find the lattice points which exist near the hyperbola $xy = N$. By this method, factors can be searched at intervals of maximally $N^{1/4}$ integers. And finaly, the possibility of parallel prosessing of the proposed factorization method with considering residue are shown.

1. Introduction

Many cipher and authentication systems base their safety on difficulty of prime factorization. Examples include RSA and Fiat-Shamir zero knowledge interactive proof . To make RSA cipher be safe, the length of the composite numbers must be shown, such that the prime factorization is computationally difficult.

The solution to the problem "what RSA block length is safe ?" depends on each prime factorization algorithm. For example, the simplest algorithm, a trial division method, requires $N^{1/2}$ trials in the worst case, where N is a composite number.

Generally, the prime factorization is said to have subexponential computational complexity. Roughly speaking, we only require $N^{1/8}$ trials to find prime factors of 200-digit composite number. Quadratic sieve method and elliptic curve method are examples of $O(N^{1/8})$ algorithm. A recent experiment shows that a 155-digit composite number is factored with the algorithms running on many computers.

In this paper, we propose a new prime factorization algorithm, which is a sort of trial division method. We define an x-y solution plane, and seek crosspoints between a hyperbola $xy = N$ and integer coordinate lattice points. To be precise, the lattice points positioned close to the hyperbola $xy=N$ are selected. For each selected

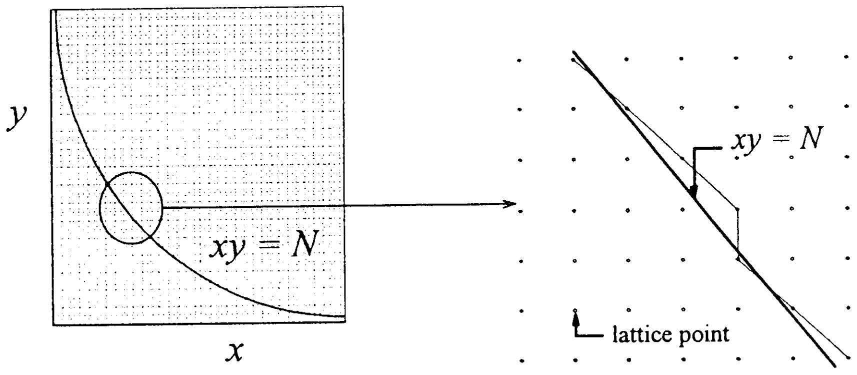

Fig. 1 Lattice points near a solution curve

lattice points (x, y), if a product of two coordinates x, y equals the composite number N, then x, y are factors of the N.

By this basic method, hereafter referred to as the lattice point search method, we can search factors at intervals of maximally $N^{1/4}$ intergers. The method is faster than the trial division method and Fermat's method. To increase the speed of the lattice point search method, we take notice of the residue of N over a prime number. From the value of residue, we can see that candidates of the factors are included in some of lattice points. Hence, we pick these points in advance, and among them, find points positioned near the hyperbola. Then, search intervals are expanded.

2. Lattice Point Search

For an arbitrary composite number N, integers x, y which solve the equation $xy=N$ are factors of N. If x (or y) is a composite number, x is factored into two integers. Continueing this method, we obtain prime factors. Hence, we only consider the problem of type $xy=N$.

2.1 Basic Concepts

Let us consider a solution plane whose coodinate axes are x and y. On this plane, there exist an infinite number of lattice points which take integers as coodinate values. Among these lattice points, we can select a sequence of points with positions near the solution curve $xy=N$. The sequence is illustrated as a line in Fig. 1. Since a large composite number, such as 10^{200}, is used for practical ciphers, the solution curve can be seen locally as if it is a straight line. Hence, if the near point line is parallel to the solution curve, two lines seldom cross on the solution plane. In other words, the crosspoints of two lines are rare. Among these points, one with positions on any of the lattice points corresponds to the factors of N.

2.2 Crosspoint Calculation

Let us define concisely the near point sequence. Firstly, two integers n and m are chosen as

$$mn > N,\ n(m-1) < N,\ n < m. \qquad (1)$$

We also define a line which passes through a lattice point (n, m), where n (m) is a value of the x (y) axis. Then, by parallel movement of the line, a set of lines is defined as

$$x = n - k \qquad (2)$$
$$y = m + \beta k + l.$$

Here, $k(\geq 0)$, l, β are integers,

$$\beta = \lfloor m/n - \alpha \rfloor \leq \alpha \qquad (3)$$

and $\lfloor x \rfloor$ is the largest integer among integers less than x. Deleting the parameter k from (2) yields

$$y = -\beta(x-n) + m + l. \qquad (4)$$

A line of $l = 0$ crosses the lattice point (n, m), and has a slant $-\beta$. Other lines of $l \neq 0$ are obtained by shifting the line of $l = 0$. These lines have the following properties.

[Theorem 1] Let us define a line P from the lines (4) of $l \leq -1$. Then, any lattice point on a line P of $x < n$ is under the solution curve, and never satisfies the relation $xy = N$.

(*proof*) Abbreviated.

[Theorem 2] Let us define a line Q from the lines (4) of $l \geq 0$. Then, for any lattice point $(x, y) = (n - k, m + \beta k)$ on a line Q, there exists k (≥ 0) which satisfies the relation $xy = N$.

(*proof*) Abbreviated.

Now, before starting the search, we select an initial lattice point with coodinate values n, m satisfying (1). Then, near the initial point, the lines (4) of $l \geq 0$ cross the solution curve $xy = N$. Among these lines, only the line of $l = 0$ has the minimal value xy, and satisfies $xy = N$. Hence, from theorem 1, 2, the line Q of $l = 0$ is the nearest to the solution curve. The crosspoint of the line Q and the curve is obtained by k_0

$$k_0 = \frac{-(m-n\beta) + \sqrt{(m-n\beta)^2 + 4\beta(nm-N)}}{2\beta} \qquad (5)$$

where k_0 is a solution of $f(k_0) = 0$.

If this crosspoint is on a lattice points, it corresponds to a factor. Otherwise, we scan the line from the crosspoint until we obtain the closest lattice point (called the bottom point). Then the next initial lattice point is determined by increasing the value of y to meet the relation (1). To summarize, a search algorithm (called a regular order search) is used as follows.

[Search algorithm 1]

[Step1] Initial lattice point (n, m) is selected to meet (1).

[Step2] Determine β from n, and (3). Calculate k_0 from (5). If k_0 is an integer, $n - k_0$, $m + \beta k_0$ are two factors.

[Step3] If k_0 has a decimal, find a bottom point by

$(x_1, y_1) = (n - \lceil k_0 \rceil, m + \beta \lceil k_0 \rceil)$

where $\lceil k_0 \rceil$ denotes a minimal integer greater than k_0.

Since at the bottom point, $xy < N$, the next initial lattice point is set to $(x_1, y_1 + 1)$, and repeat **[Step 2]**.

3. Analysis of Search Intervals

The interval k_0, the interval of search argolithm 1, has a maximal value when $m \underset{-}{\sim} \beta n$ or, in other words, when m is integer times larger than n. Then the value of maximal k_0 depends on values of $nm - N$ and β from (6).

$$1 \leq \lceil k_0 \rceil \leq \left\lceil \sqrt{\frac{nm-N}{\beta}} \right\rceil \qquad (6)$$

Of these values, $nm - N$ can be geometrically estimated. When the initial lattice points n, m satisfy $m \underset{-}{\sim} \beta n$, line Q and the solution curve become parallel as stated before . Hence, a bottom point (n', m'), near the crosspoint of line Q and the curve, exists as if it were on the curve. This yields $n'm' \underset{-}{\sim} N$. Hence, for the next initial lattice point $(n', m' + 1)$

$$n'(m'+1) - N \underset{-}{\sim} n \qquad (7)$$

holds. Though the initial condition (1) only says that $nm < N$, the relation $nm - N \underset{-}{\sim} n$ holds as long as $m \underset{-}{\sim} \beta n$ is satisfied.

For the value $\beta, \beta \underset{-}{\sim} 1$ when n, $m \underset{-}{\sim} N^{1/2}$, and β increases when n decreases. The maximal search intervals (when $m \underset{-}{\sim} \beta n$ stands) is calculated as a function of n from (6), and are shown in Fig. 2.

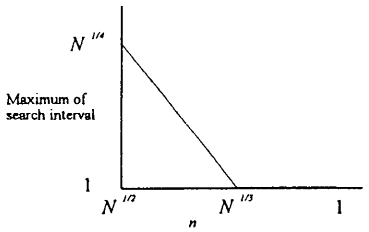

Fig. 2 Search interval and search position

From this, the interval has maximal value $N^{1/4}$ when n, $m \simeq N^{1/2}$. However, it becomes smaller as n decreases. Hence, this method is basically effective when factors exist in the neighborhood of $N^{1/2}$. Suppose N equals 10^{200}.

$n = 10^{100}$: interval $= 10^{50}$

$n = 10^{98}$: interval $= 10^{47}$

....

$n = 10^{66}$: interval $= 1$

hold. For n satisfying $10^{98} \leq n \leq 10^{100}$, which occupies 99 % of the total search area, the search area, the search proceeds at the interval of 10^{47} (if m is integer times larger than n). On the other hands, for small n, the search proceeds at the interval 1 in the worst case. Hence, more than 10^{66} searches are required.

When $m/n \simeq \beta$ does not hold, m/n has a decimal, and the search proceeds. Noted at (6), every lattice point near the solution curve must be examined in the worst case. How to improve the algorithm in such a case is discussed later(3.3 Virtual Lattice point).

[Example 1] When $N = 1.0 \times 10^{11}$, the search intervals k_0 of algorithm 1 are shown in Fig. 3. A horizontal axis denotes n. The search starts at point A, where $m = n = N^{1/2}$, and proceeds so that n may decrease. At point B, $m = 2n$ holds, and the search intervals increase. At point C, n becomes $N^{1/2}$, and the search intervals become shorter as shown in Fig. 2. However, the area $1 \sim N^{1/3}$ occupies only a small portion of the total search area $1 \sim N^{1/2}$.

3.2 Inverse Order Search

When the initial lattice point has the coordinate values (n, m), with m/n slightly smaller than an integer value (*e.g.* m/n = 1.99), algorithm 1 has small search intervals. However, searching both regular and inverse directly, we can make the search intervals large.

3.3 Virtual Lattice Point

Even if algorithm 1, 2 are used, there remains a large area where the search intervals are small. In the area, the coordinate ratio m/n of the initial lattice point has decimal values. To improve the search intervals in such cases, we introduce virtual lattice points.

For example, when $m/n \simeq 1.5$, lattice points shifted by 1/2 in the direction of the y axis are added to the original lattice points. Then, line Q which passes through the lattice points near the solution curve, becomes parallel to the solution curve (actually the tangent of the curve). Hence, large search intervals are attained. The difference from the above algorithm is that a crosspoint on a virtual lattice point does not coordinate values are integers.

Generally, when $m/n \simeq R + t/s$ (R, s, and t are integers), lattice points, shifted by w/s (w = 1, 2, ... , $s-1$) in the direction of the y axis, are added to the original lattice points. We denote a set of these lattice points as W. We will explain the details of the regular order search as explained in algorithm 1.

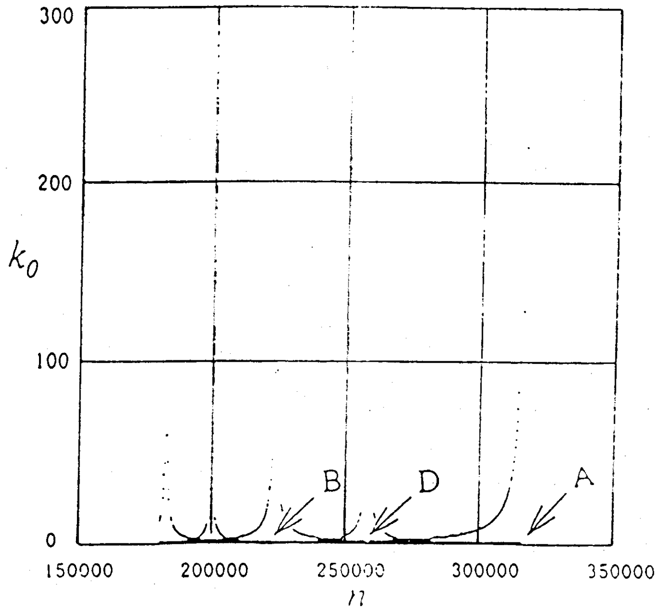

Fig. 3 k_0 and n of searching algorithm 2(N=10^{11})

In the point set W, the initial lattice point is selected as

$nm > N, n(m-1/s) < N,$

$R + t/s \leq m/n < R + (t+1)/s$ (8)

(n, R, and s are integers.)

are satisfied. A set of lines, which linl the points of W near the solution curve, are formalized as

$x = n - k$ (9)

$y = m + (R + t/s)k + l/s$ (l = 1, 2, ...)

and theorem 1, 2 are proved.

In (9), a crosspoint of the line Q, with l = 0 and the solution curve is determined by k_0, where k_0 satisfies $f(k_0) = N - xy = 0$, and given by

$$k_0 = \frac{-[m - nR_1] + \sqrt{[m - nR_1]^2 - 4R_1(N - nm)}}{2R_1(R + t/s)}$$

$$R_1 = (R + t/s) \tag{10}$$

[Search algorithm 2]

[Step 1] Initial lattice point (n, m) is selected to meet (8).

[Step 2]Calculate k_0 from (10). Find a bottom point by $(x_1, y_1) = (n - \lfloor k_0 \rfloor, m + (R + t/s)\lfloor k_0 \rfloor)$.

[Step 3] If $x_1 y_1 = N$,

if y_1 is an integer, x_1, y_1 are facters,

if y_1 is a decimal, set the next initial lattice point to (x_1, y_1 + $1/s$),

and repeat **[Step 2]**.

[Step 4] If $x_1 y_1 < N$, find the minimal v satisfying $x_1(y_1 + v/s) \geq N$,

[Step 5] if $x_1(y_1 + v/s) = N$,

if $y_1 + v/s$ is an integer, x_1, $y_1 + v/s$ are factors,

if $y_1 + v/s$ is a decimal, set the next initial lattice point

to (x_1, $y_1 + v/s + 1/s$), and repeat **[Step 2]**,

[Step 6] if $x_1(y_1 + v/s) > N$, set the next initial lattice point to (x_1, $y_1 + v/s$), and repeat **[Step 2]**. □

4 Consideration on Residue

The preceding algorithm has maximally $N^{1/4}$ search intervals. In the case of RSA, the composite number N is nearly 10^{200}, and $N^{1/4}$ is 10^{50}. though it is a large interval, it requires more than 10^{50} searches to cover the total search area $N^{1/2}$.

Hence, whether or not a higher algorithm exists is important. To clarify this problem, let us devide the composite number N by a prime number, and obtain a residue. From the residue, we can estimate a class of lattice points. Then we can construct a similar algorithm on the extracted lattice points.

4.1 Residue Class

For a prime number p, there exist p residue classes Z_0, Z_1, Z_2, ... , Z_{p-1}. Here, Z_i is defined as a set of integers z's such that $z = i \bmod p$. Suppose the composite number N is expressed as a product of two integers. Then, each integer belongs to one of the residue classes. Generally, a pair of residue classes which generate N is limited to the following forms,

$$\left.\begin{array}{l} N = z_1 z_j \\ N = z_2 z_k \\ \cdots \\ N = z_{p-1} z_1 \end{array}\right\} p-1 \text{ forms}\left(z_t \in Z_t\right). \quad (11)$$

For example, if N mod 3, N is divided into one of the following forms:

$$N = (3a + 1)(3b + 1) \quad (12)$$
$$N = (3a + 2)(3b + 2)$$ where a, b are integers.

4.2 Computational Quantity for Search

Suppose one of the forms is selected from (9). Let us denote it as

$$N = z_i z_j \left(z_i \in Z_i, z_j \in Z_j\right). \quad (13)$$

Then in algorithm 1, the initial lattice point is chosen as

$$nm > N,\ n(m-p) < N \quad (14)$$
$$n < m,\ \left(n \in Z_i, m \in Z_j\right).$$

A set of lines, corresponding to (2) are

$$x = n - pk \quad (15)$$
$$y = m + p\beta k + pl$$

and with these lines, theorem 1, 2 hold. The value k_0, relating to (5), is calculated from

$$(n - pk_0)(m + p\beta k_0) = N$$

and

$$k_0 = \frac{-(m - \beta n) + \sqrt{(m - \beta n)^2 + 4\beta(nm - N)}}{2p\beta} \quad (16)$$

Then the search intervals satisfy

$$1 \le \lceil pk_0 \rceil \le \left\lceil \sqrt{\frac{nm - N}{\beta}} \right\rceil \quad (17)$$

(righthand equality holds when $m = \beta n$).

Equation (15) has the same expression as (6), but the values inside the root symbol differ from each other. To clarify the discussion, we firstly propose that the slant $-\beta$ of lines (13) is almost equal to the slant of the solution curve. This is the essential condition for enlarging the search intervals, and is realizeable with virtual lattice points as stated bofore. Then, the bottom point positions are near the solution curve. The next initial lattice point (n, m) has a value m larger than that of the bottom point, and the difference almost equals a lattice interval. Since the residue of mod p is considered, the lattice interval is p times larger than when the residue is not considered. Hence from (15), the search intervals pk_0 are $\sqrt{p}$ times larger than (6).

Therefore, taking account of residue class, the search trials are reduced to $1/\sqrt{p}$ compared with the ordinary search on integer lattices. However, a pair of residue classes has $p-1$ forms as stated in (9). Since specifing a true form among $p-1$ forms is difficult, the total number of searches increases by $(p-1)/\sqrt{p}$ times. We now present an example to overcome this problem.

[Example 2] Let us consider a composite number $N = 1315753$ and modulo $p = 6$.

$$N \equiv 1 \pmod 6 \qquad (18)$$

then N is factored as next forms:

$$N = (6a+1)(6b+1), \qquad (19)$$

or $N = (6a+5)(6b+5)$.

(a and b are positive integers.) (20)

Now, assume N is denoted as equation (19). Hence, the factors are included in series:

1,7,13,19,25,31,37,43,49, ...

while, the LSD of N is 3 (Condition 1.). Then, factors are

1,7,13,19,31,37,43,49, ...

Since equation (16),

$36ab+6a+6b=1315753$

$6ab+a+b=219292$

$$a+b \equiv 4 \pmod 6 \qquad (21)$$

Then the candidates are

4,10,16,22,28,34,40,46,52,58, ...

This result is shown as the following graph Fig. 4.

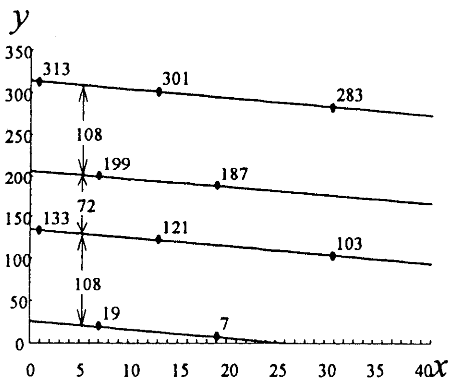

Fig. 4 Candidates of factor in Example 2

As mentioned above, the results are shown in the following tables. In table 1, the set of candidates of mod 4 are the subset of mod 2. Similarly, candidates of mod 6 are the subset of candidates mod 3. Hence, it is possible to expand a search interval if we select large composite modulus.

Table 1 shows average interval of lines which pass though candidates of factor (illustrated in Fig. 4). Since search intervals are increased by $\sqrt{p}$ times when intervals of lines become p times larger. Hence, for example, search interval in the case of mod 12 is 11 times ($\simeq\sqrt{124}$) larger than original search interval.

Table 1. The relation between modulus and the search intervals

mod p	Period	Num of elements	Average interval
2	20	2	10
3	90	4	22.5
4	240	10	24
9	22680	240	94.5
•	•	•	•
12	774400	6201	124.88
•	•	•	•
20	2293200	30038	76.34

Table 2. The relation between combination of modulus and the search intervals

Combination of modulus	Period	Num of elements	Average interval
2,3	180	4	45
2, 3, 5	900	19	47.37
2, 3, 5, 7	44100	420	105
3, 5, 8	100800	479	210.44
3,8	20160	95	212.21
8,9	181440	312	581.54

5. Conclusion

We proposed a new prime

factorization algorithm, which seeks factors of a composite number N by finding the crosspoint of solution curve $xy = N$ and a line made of lattice points near the curve. In this algorithm, when a slant of the curve (to be precise, a tangent of the curve) equals a slant of the line, the search progresses by the order of $N^{1/4}$. In other cases, however, the search intervals decrease. Hence, we proposed a method to make the two lines parallel by adding virtual lattice points whose coordinates are decimals.

To consider the practical meaning of algorithm, suppose the algorithm is used to decipher RSA. Since RSA uses a large composite number, such as 10^{200}, the search interval of this method grows to maximally 10^{50}. Hence, it is necessary, in future, to speed up the proposed algorithm. To do this, we proposed a method to specify the residue expression of the composite number.

The presented algorithm is experimentally examined, and resulting numerical data are shown in this paper. With this examination, we can confirm that the basic idea, presented here, is true. However, an accumulation of data, especially on a large composite number, remained for future research. Finally, we comment on the parallel processing of the algorithm. Since the initial lattice points can easily be calculated from (1) etc., the proposed algorithm is suited to parallel processing like the other factoring algorithm.

References

[1]R.Rivest, A.Shamir and L.Adleman "*A method for obtaining digital signatures and public-key cryptosystems*", Comm. of ACM, pp.120-126(Feb. 1978)

[2]Hiroshi NAGASE, Naoki TAKEDA "*On the selection of Public Modulus for RSA Cipher*"

[3]Shi'nichi.SHIMONAKA, Naoki.TAKEDA and Hiroshi NAGASE "*Parallel Decomposition of Modular Exponentiation for RSA Cryptosystem*", ISITA 1994

eds.), Math. Centrum Tracts, Number 154, Part I and Number 155, Part II, Amsterdam, pp.89-139 (1983)