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

2007－ARC－172 (35)
2007－HPC－109 (35)
2007／3／2

大規模素因数分解のための高性能計算環境の実現

西田 晃[†,††]

今日の電子社会は高度な暗号・検証システムに支えられているが，その安全性を保証するためには，高性能な計算機環境を用いた十分な検証が必要である．本稿では，公開鍵暗号においてその計算量が問題となる大規模素因数分解に必要な計算機性能について考察し，この計算に適した計算機システムを提案する．

Building High Performance Computing Environment for Large Scale Factoring

Akira Nishida[†,††]

Today's cyber society is dependent on high level cryptographic authentication systems, while sufficient verification with advanced computing resources is necessary to guarantee their security. In this paper, we analize the performance required for large scale factorizations, and propose an appropriate system for such computation.

1. はじめに

今日の電子社会は高度な暗号・検証システムに支えられている．暗号の安全性を保証するためには，高性能な計算機環境を用いた十分な検証が必要であるが，国内では理論的な評価が中心であり，暗号・認証技術の安全性を迅速に検証できる体制が確立されているとは言いがたい．本研究では，数論アルゴリズムとその実装に関する研究，全国共同利用施設等を活用した大規模な計算処理能力，多様なハードウェア技術等を統合することにより，総合的な研究体制を推進することを目指している．本稿では，公開鍵暗号においてその計算量が問題となる大規模素因数分解に必要な計算機性能について考察し，この計算に適した計算機システムを提案する．

2. 背景

不特定多数の利用者が存在するネットワーク上において安全に通信を行う上で，公開鍵暗号は現在もっとも重要な要素技術である．公開鍵暗号では，暗号化に必要となるすべての情報は公開され，だれでも暗号化を行うことができる一方，作成された暗号文は，秘密に保管されている復号鍵を用いない限り復号することができない．

公開鍵暗号方式は，鍵生成アルゴリズム GEN，暗号化アルゴリズム ENC，復号アルゴリズム DEC の組として定義される．

(1) GEN: 1^k を入力とし，復号鍵と公開鍵のペア (sk, pk) を出力する確率的アルゴリズム．ここで k はセキュリティパラメータとする．

(2) ENC: 平文 $m \in M$ と公開鍵 sk を入力とし，暗号文 c を出力する確率的もしくは確定的アルゴリズム．

(3) DEC: 暗号文 c と復号鍵 pk を入力とし，平文 m もしくは失敗を表すシンボル $\perp$ を出力する確定的アルゴリズム．

受信者は GEN により鍵ペアを生成し，このうち公開鍵を周知にする．送信者は ENC により平文を受信者の公開鍵で暗号化し，受信者に送信する．受信者は DEC により，復号鍵を用いて暗号文の復号を行う．

2.1 RSA 暗号

RSA 暗号は，現在最も広く利用されている公開鍵暗号方式である．以下に RSA 暗号の構成を示す．

(1) 鍵生成: 入力 1^k に対し，二つの素数 p, q を生成し，$n = pq$ を計算する．ここで，$|p|$ と $|q|$ は，n の素因数分解が困難となるようなほぼ等しい値とする．また，$\lambda(n) = \mathrm{lcm}(p-1, q-1)$ を計算する．さらに，適当な $e \in Z_{\lambda(n)}$, $\gcd(e, \lambda(n)) = 1$ を定め，$ed = 1 \bmod \lambda(n)$ となるような d を導出する．復号鍵，公開鍵のペアとして，次のように

† 中央大学21世紀 COE プログラム
21st Century COE Program, Chuo University
†† 科学技術振興機構 CREST
CREST, JST

(sk, pk) を出力する.

$$\mathrm{sk} = (d, n), \mathrm{pk} = (e, n) \quad (1)$$

(2) 暗号化: 入力 $m \in Z_n$ 及び pk に対し, 次式により暗号文 c を得る.

$$c = m^e \bmod n \quad (2)$$

ここで, m は暗号化の対象となる平文とする.

(3) 復号化: 入力 $c \in Z_n$ 及び sk に対し, 次式により復号を行う.

$$m = c^d \bmod n \quad (3)$$

RSA 暗号の公開鍵 (e, n) と, $y \in Z_n$ が与えられたときに, $x = y^{1/e} \bmod n$ を満たす x を求める問題を, RSA 問題という. n の素因数分解を行うことなく RSA 問題を解く方法は知られていないため, これを破るための最も効率的な手段は, n の素因数分解であると考えてよい.

3. 素因数分解

整数 n を素因数分解するアルゴリズムは2種類に大別される. 一つは, n に含まれる最小素因数 f の大きさに依存して実行時間が決まるものであり, 他方は n の大きさだけに依存して実行時間が決まるものである. 前者に属するものとして, 試行割算法, Pollard 法 (ρ 法), 楕円曲線法 (ECM) などを, また後者に属するものとして, Lehmann 法, 連分数法, 2次篩法, 複数多項式2次篩法 (MPQS), 一般数体篩法 (GNFS) などを挙げることができる[7),9),10)]. これらのアルゴリズムの実行時間は, いずれも入力サイズの準指数関数オーダである.

最小素因数の大きさに依存するもののなかで最速のアルゴリズムは ECM である. ECM では, 体上で定義される楕円曲線を環 Z/nZ の上で定義し, 曲線の次数が n の最小素因数の倍数になるような曲線を発見することで, n を分解する. ECM に関する計算量は, $c \approx 2$ を定数として, $O(\exp(\sqrt{c \ln f \ln \ln f} \cdot (\log n)^2)$ と書けることが分かっているので, ムーアの法則を考慮すれば, n の10進桁数 D に対して, 解読までにおおよそ $D^{1/2}$ ($\sqrt{\ln D}$ は定数とみなす) に比例する時間がかかると考えることができる. 実際, 1991年から1999年までの記録をもとに

$$Y = 9.3\sqrt{D} + 1932.3 \quad (4)$$

とすると, $D = 60$ の場合に $Y \approx 2004$ となる.

一方, 整数 n の大きさだけに依存して実行時間が決まるもののなかで最速のアルゴリズムは GNFS である. GNFS では, $k^2 \equiv l^2 \bmod n$ なる $k, l, k \neq \pm l$ を発見する2次篩法を代数的整数上に拡張する. そのような k, l を発見できれば, $(k-l)(k+l) = 0 \bmod n$ により, $\gcd(k \pm l, n)$ が n の自明でない因数を与える.

3.1 GNFS

GNFS の概要は以下の通りである. 今 n を合成数, $f \in Z/nZ[X]$ を k 次既約モニック多項式とする. m を整数とし, $n = f(m)$ を分解する. $f = 0$ の根の一つを $\alpha \in C$ とする. 代数体 K を $K = Q(\alpha)$ とし, O_k を K の整数環とする. $O_k = Z[\alpha]$ は素元分解環とする. まず, 因子基底 B_Q, B_K を用意する. B_Q は上限 B より小さい有理素数の集合, B_K は B より小さいノルムを持つ O_K の一次素イデアル及び O_K の単元の基本集合とする. ここで $c + d\alpha \in O_K$ のノルム $N(c + d\alpha)$ は $N(c + d\alpha) = |(-d)^k f(-c/d)|$ によって計算される. 具体的な手順は以下の通りである.

(1) (c_i, d_i) を互いに素な有理整数とする. $c_i + d_i m$ は B_Q スムーズであり, かつ $c_i + d_i\alpha$ が B_K スムーズとする. $c_i + d_i m$ 及び $c_i + d_i\alpha$ が完全に分解されるようなペア (c_i, d_i) を多く集める.

(2) $\Pi(c_i + d_i m) = s^2$, $\Pi(c_i + d_i\alpha) = t^2$ を満たす (c_i, d_i) の集合を見つけ, $\gcd(s \pm t, n)$ を計算して n の素因数を求める.

(1) は関係式収集ステップ, (2) は行列計算ステップとも呼ばれ, 両者で GNFS の計算の大部分を占める. 関係式収集ステップは多数の計算機による分散処理が可能であるが, 行列計算ステップについては, 一般に高性能なネットワークを備えた大規模並列計算環境が必要となる.

GNFS に関する計算量は, 合成数を n, c を適当な定数として $O(\exp(c(\ln n)^{1/3}(\ln \ln n)^{2/3}))$ と書けることが分かっているので, ムーアの法則を考慮すれば, n の10進桁数 D に対して, 解読までにおおよそ $D^{1/3}$ ($(\ln D)^{2/3}$ は定数とみなす) に比例する時間がかかると考えることができる. 実際, 1964年から2000年までの記録をもとに

$$Y = 13.24D^{1/3} + 1928.6 \quad (5)$$

とすると, $D = 200$ の場合 $Y \approx 2006$ となるが, この問題は昨年 GNFS により解かれているので, ほぼ正確に成り立っていると考えてよい.

4. アルゴリズム

以下では上記のアルゴリズムの詳細について述べる. GNFS のアイデアは2次篩法に基づいているため, ここでは2次篩法の場合について考察する.

2次篩法では, n の素因数を見つけるために, $x \neq \pm y \pmod n$ かつ $x^2 = y^2 \pmod n$ なる2整数 x, y を求める. このとき $\gcd(x - y, n)$ は n の因数を与える. x, y を求めるため, 適当な大きさの素数の集合, すなわち因子基底 (factor base) を選び, これらの積から x, y を構成することにする.

例えば $n = 1098413$ を因数分解する場合, 関数 $f(x) = x^2 - n$ に x として $\sqrt{n} \approx 1048$ に近い

数を代入し，平方非剰余を除いた因子基底を例えば $S = \{2, 7, 13, 17, 19, 23\}$ として，この上で $f(x)$ の素因数分解を試みる．この場合，

$$f(1051) = 2^2 \cdot 7 \cdot 13 \cdot 17 \quad (6)$$
$$f(1063) = 2^2 \cdot 7^3 \cdot 23 \quad (7)$$
$$f(1077) = 2^2 \cdot 7 \cdot 13^3 \quad (8)$$
$$f(1119) = 2^2 \cdot 7 \cdot 17^2 \cdot 19 \quad (9)$$
$$f(1142) = 7^2 \cdot 13 \cdot 17 \cdot 19 \quad (10)$$
$$f(1237) = 2^2 \cdot 13 \cdot 19^2 \cdot 23 \quad (11)$$

となるので，S の因子に行を対応させ，因子のべき指数が奇数となる場合に要素 1 を与えた行列

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (12)$$

を書き出すことができる．このとき，第2,3,6列は線形従属である．(実際，$x^T = (011001)$ ならば $Ax = 0$ となる．) したがって

$$(1063 \cdot 1077 \cdot 1237)^2 = (2^3 \cdot 7^2 \cdot 13^2 \cdot 19 \cdot 23)^2 \bmod n \quad (13)$$

が成り立つので，

$$326330^2 = 391638^2 \bmod n \quad (14)$$

より

$$\gcd(391638 - 326330, n) = 563 \quad (15)$$

として素因数563が求まる．ただし，第1,4,5列を選んだ場合のように，

$$810112^2 = 810112^2 \bmod n \quad (16)$$

となって分解に寄与しないこともあることに注意する．

MPQS, GNFS においても同様の理由で，関係式の収集と (2を法とする有限体 GF(2) 上での線型方程式 $Ax = 0$ の求解により) 列ベクトル間の依存性を見つける作業が必要となる．

関係式収集ステップに関して特異な点は以下の通りである．

(1) 互いに依存関係のない大量の小規模な素因数分解に帰着されるため，通信が発生せず，並列処理が容易である．
(2) 空間局所性が高い．

一方，行列計算ステップで現れる係数行列は一般に疎となるため，反復解法が用いられることが多い．GF(2) 上での処理において特異な点は以下の通りである．

(1) 非零ベクトルは自分自身と直交する場合がある．この場合，ブレイクダウンを防ぐための処理が必要となる．
(2) 正確な解が必要となるため，近似的に計算する場合に比べて反復回数が多くなる．
(3) 前処理が有効に働かない．
(4) 行列は一般に非対称である．
(5) GF(2) 上の計算は，ワード上のビット演算として扱うことにより，並列化が可能である．

(1) の問題に関しては，後に述べるブロック Lanczos 法が Montgomery[8] らにより提案されている．

$Rx = 0$ を満たす $x \neq 0$ を求めることとすると，適当な乱数ベクトル c を選んで，$b = Rc$ とし，$Ry = b$ を解くと，$y - c$ は条件を満たす．したがって，以下では $Ry = b$ を解くことを考える．

Lanczos 法は対称正定値行列に対して適用されるので，ここでは $Rx = b$ の代わりに正規方程式

$$R^T Rx = R^T b \quad (17)$$

を解く．A は疎行列性を失うが，$Ax = R^T Rx$ は $R^T(Rx)$ から求めることができ，実際の計算には影響しない．$R^T z$ も $(z^T R)^T$ から計算できるので，転置行列の計算は不要である．

これらの計算に前処理を適用することはできないが，事前に Gauss の消去法を数ステップ実行することにより，行列の次数を小さくすることは可能である．実際，行列 A の列に，1つだけ非零要素を含む列が含まれているとすると，$Az = 0$ ならば対応する z の要素は零となるので，この列は線形従属性の判定に寄与しない．したがって，この列と，対応する行を消去することができる．

5. 計算環境

ここでは，大規模素因数分解のための計算環境を現実的なコストで実現することを考える．

まず，関係式収集ステップについては，空間局所性が高いことから，キャッシュサイズの大きなアーキテクチャが有利であると考えられる．また，メモリ帯域幅をそれほど要しないことから，CPU 当たりのコア数が大きなアーキテクチャであっても問題は生じにくい．メモリ帯域幅に対する要求がそれほど大きくないことから，比較的安価なメモリを大量に利用することが望ましいと考えられる．なお，関係式収集に関して Intel Core 2 Duo サーバとシングルコア AMD Opteron 2P サーバ上で予備的な調査を行った結果，クロック差を補正した場合のメモリ帯域幅による差異，複数のプロセスを実行することによる性能低下は見られなかった．

一方，行列計算ステップでは疎行列を扱うため，空間局所性が小さく，帯域幅に敏感である．計算ノード内のメモリ帯域幅，計算ノード間の通信帯域幅を要し，計算機環境に必要な要件が関係式収集部と異なっている点に留意する必要がある．

以上の観点から，本研究では，関係式収集と行列計算に関して，それぞれに適した計算環境を構築することとし，仕様を決定した．

まず，関係式収集ステップに関しては，上記の条件

から, Intel 社の Core アーキテクチャを利用した PC クラスタとすることにした. Core アーキテクチャでは, コア当たり L1 キャッシュを 32KB, L2 キャッシュを 2MB 持ち, 4コアを1パッケージにおさめた製品が既に出荷されている (図1). メモリについては, 最も安価なサーバでも2GB の DDR メモリと併用することにより, 8GB まで搭載することができる. また, ノード間通信についてはそれほどの性能を要しない. したがって, これらの条件を組み合わせることにより, 最適な計算環境を構築することができる. 本研究では, Xeon 3220 (quad core, 2.4GHz) を 1P サーバ用のマザーボード Supermicro PDSMI+ と組み合わせて, 8GB までメモリを搭載した計算ノード16台をギガビットイーサネットで接続することとし, 現在環境を構築している段階である.

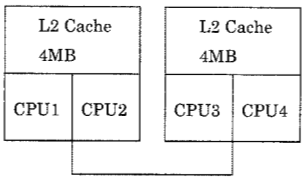

図1 Intel Xeon 3220 プロセッサダイアグラム

また, 行列計算ステップに関しては, 同様に上記の条件から, AMD 社の Opteron プロセッサを利用した PC クラスタとすることにした. Opteron はコア当たり最大 6.4GB/s のメモリ帯域幅を持ち, PCI Express と InfiniBand 等の高速インターコネクト技術を併用することにより, ノード間の通信帯域幅も合わせて確保することができる. 本研究では, Opteron 248 (single core, 2.2GHz) を 2P サーバ HP DL145 G2 と組み合わせ, 2GB までメモリを搭載した計算ノードを16台導入し, DDR InfiniBand ネットワークで接続している (図2).

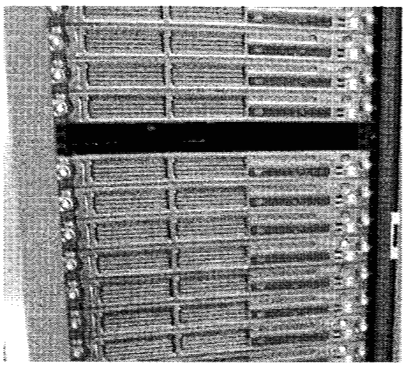

図2 行列計算ステップ用クラスタ環境

6. ま と め

本稿では, 大規模素因数分解に計算量が問題となる関係式収集ステップ及び行列計算ステップのそれぞれに必要な計算機性能について考察し, 事前の評価結果から関係式収集ステップに関してはメモリ性能が重要となること, 行列計算ステップに関しては高速なネットワークの使用が望ましいことなどを示すとともに, これらの計算に適した計算機システムを提案した.

謝辞 本研究の一部は, 中央大学21世紀 COE プログラム「電子社会の信頼性向上と情報セキュリティ」, 科学研究費補助金若手研究 (A) 17680001, 及び科学技術振興機構戦略的創造研究推進事業によるものである.

参 考 文 献

1) Aoki, K. and Ueda, H.: Sieving Using Bucket Sort., *ASIACRYPT*, pp. 92–102 (2004).
2) Brent, R. P.: Recent Progress and Prospects for Integer Factorization Algorithms, *LNCS*, Vol. 1858, pp. 3–22 (2000).
3) Coppersmith, D.: Solving linear equations over GF(2): Block Lanczos Algorithm, *Lin. Alg. Appl.*, Vol. 192, pp. 33–60 (1993).
4) Coppersmith, D.: Solving linear equations over GF(2) via block Wiedemann algorithm, *Math. Comp.*, Vol. 62, No. 205, pp. 333–350 (1994).
5) LaMacchia, B. A. and Odlyzko, A. M.: Solving Large Sparse Linear Systems over Finite Fields, *LNCS*, Vol. 537, pp. 109–133 (1991).
6) Lambert, R.: *Computational aspects of discrete logarithms*, PhD Thesis, University of Waterloo, Canada (1996).
7) Lenstra, A. and Lenstra, H.: The development of the number field sieve, *LNCS*, Vol. 1554 (1991).
8) Montgomery, P. L.: A Block Lanczos Algorithm for Finding Dependencies Over GF(2), *LNCS*, Vol. 921, pp. 106–120 (1995).
9) Riesel, H.: *Prime Numbers and Computer Methods for Factorization*, Birkhaeuser (1994).
10) 岡本龍明 (編): 数論アルゴリズムとその応用, 情報処理, Vol. 34, No. 2 (1993).
11) 西田晃: 大規模素因数分解のための GF(2) 上疎行列線型方程式解法の性能解析, 2007年暗号と情報セキュリティシンポジウム, CDROM (2007).
12) 電子情報通信学会 (編): 情報セキュリティハンドブック, オーム社 (2004).
13) 木田祐司: 暗号アルゴリズムの詳細評価に関する報告書, 技術報告, 情報処理推進機構 (2002).