高速乗算法の設計と実装 (1)

梅谷 武

作成：2001-08-25 更新：2005-04-20

ショーンハーゲ-ストラッセン法を俯瞰することによって、多倍長整数の高速乗算法の構造についてより抽象的な視点で検討し、その応用として 32bit 算術演算器向け高速乗算法を設計し、Pentium への実装を試みる。
IMS:20010825001; NDC:412.1; keywords:離散 Fourier 変換, 数論変換;

目　　次

1　高速乗算法の構造

1.1　ショーンハーゲ-ストラッセン法の構造

ショーンハーゲ-ストラッセン法の概略を少し改良した形で述べ、その全体構造を俯瞰する。

数の P 進表現と積の評価

$K = 2^k (k > 0), N = 2^K$ とし、正の整数 a, b が基数 $P = 2^K$ により次のように P 進表現されるものとする。

$$a = a_{K-1}P^{K-1} + \cdots + a_1 P + a_0, 0 \leqq a_i < P$$
$$b = b_{K-1}P^{K-1} + \cdots + b_1 P + b_0, 0 \leqq b_i < P$$

この場合 $0 \leqq a, b \leqq 2^N - 1$ であるが、この積について $0 \leqq ab \leqq 2^N - 1$ となるように N を定める。このとき ab は $2^N - 1$ を法として求めればよい。

多項式としての積とその係数の評価

$P^K \equiv 1(\mathrm{mod}\, 2^N - 1)$ より、

$$c = ab \equiv \sum_{r=0}^{K-1} (\sum_{s+t\equiv r(modK)} a_s b_t) P^r (\mathrm{mod}\, 2^N - 1)$$

となる。この係数 $c_r = \sum_{s+t\equiv r(modK)} a_s b_t$ の大きさを評価すると、

$$0 \leqq c_r \leqq K(2^K - 1)^2 < K(2^{2K} + 1)$$

となる。したがって c_r は $K(2^{2K} + 1)$ を法として求めればよい。

孫子の剰余定理による係数の計算

$(K, 2^{2K} + 1) = 1$ であるから、孫子の剰余定理から c_r を、c'_r 及び c''_r が既知であるとして、連立合同式

$$c_r \equiv c'_r (\mathrm{mod}\, K)$$
$$c_r \equiv c''_r (\mathrm{mod}\, 2^{2K} + 1)$$

を解くことによって求めることができる。

離散フーリエ変換による係数の計算

c'_r は直接計算し、c''_r は、$(K, 2^{2K} + 1, 2^4)$ 型の離散 Fourier 変換を利用して次のように計算する。

$$F(a)_i \equiv \sum_{s=0}^{K-1} a_s \zeta^{si} (\mathrm{mod}\, 2^{2K} + 1)$$
$$F(b)_i \equiv \sum_{t=0}^{K-1} b_t \zeta^{ti} (\mathrm{mod}\, 2^{2K} + 1)$$
$$c''_r \equiv K^{-1} \sum_{i=0}^{K-1} F(a)_i F(b)_i \zeta^{-ir} (\mathrm{mod}\, 2^{2K} + 1)$$

桁上げ処理

最後に $c_r, r = 0, \cdots, K-1$ に桁上げ処理を施すことで、$c = ab$ の P 進表現が得られる。

1.2 高速乗算法の構造

ショーンハーゲ-ストラッセン法の構造からわかることは、高速乗算法が次のような**乗算変換**を組み合わせてできあがっているということである。(→ [N5],[D2], [D3])

定義 1.1 (乗算変換) 可換環 R から可換環 R' への準同型写像を

i. ある元 p が生成するイデアルにより剰余環 $R' = R/(p)$ と自然な準同型写像 $R \to R' = R/(p)$
ii. 何らかの同型定理に基づく同型写像 $R \to R'$

という 2 つの方法のどちらかで作成し、可換図式

$$\begin{array}{ccc} R \times R & \to & R \\ \downarrow & & \downarrow \\ R' \times R' & \to & R' \end{array}$$

によって R の乗算を R' の乗算に帰着させることを**乗算変換**と呼ぶ。

ショーンハーゲ-ストラッセン法を構成する乗算変換を列挙してみる。

乗算変換１

自然な準同型写像

$$\mathbf{Z} \to \mathbf{Z}/(2^N-1)\mathbf{Z}$$

による乗算変換

$$\begin{array}{c} \mathbf{Z}\times\mathbf{Z}\to\mathbf{Z} \\ \downarrow \quad \downarrow \\ \mathbf{Z}/(2^N-1)\mathbf{Z}\times\mathbf{Z}/(2^N-1)\mathbf{Z}\to\mathbf{Z}/(2^N-1)\mathbf{Z} \end{array}$$

乗算変換２

自然な準同型写像

$$\mathbf{Z}[x] \to (\mathbf{Z}/K(2^N-1)\mathbf{Z})[x]$$

による乗算変換

$$\begin{array}{c} \mathbf{Z}[x]\times\mathbf{Z}[x]\to\mathbf{Z}[x] \\ \downarrow \quad \downarrow \\ (\mathbf{Z}/K(2^{2K}+1)\mathbf{Z})[x]\times(\mathbf{Z}/K(2^{2K}+1)\mathbf{Z})[x]\to(\mathbf{Z}/K(2^{2K}+1)\mathbf{Z})[x] \end{array}$$

乗算変換３

孫子の剰余定理による同型写像

$$\mathbf{Z}/K(2^{2K}+1)\mathbf{Z} \to \mathbf{Z}/K\mathbf{Z}\times\mathbf{Z}/(2^{2K}+1)\mathbf{Z}$$

による乗算変換

$$\begin{array}{c} \mathbf{Z}/K(2^{2K}+1)\mathbf{Z}\times\mathbf{Z}/K(2^{2K}+1)\mathbf{Z}\to\mathbf{Z}/K(2^{2K}+1)\mathbf{Z} \\ \downarrow \quad \downarrow \\ \mathbf{Z}/K\mathbf{Z}\times\mathbf{Z}/(2^{2K}+1)\mathbf{Z}\times\mathbf{Z}/K\mathbf{Z}\times\mathbf{Z}/(2^{2K}+1)\mathbf{Z}\to\mathbf{Z}/K\mathbf{Z}\times\mathbf{Z}/(2^{2K}+1)\mathbf{Z} \end{array}$$

乗算変換４

$G=\zeta^i|i=0,1,\cdots K-1$ としたとき離散フーリエ変換の構造定理 (→ [N3]) による同型写像

$$(\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[x]/(x^K-1) \to (\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[G]$$

による乗算変換

$$\begin{array}{c} (\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[x]/(x^K-1)\times(\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[x]/(x^K-1)\to(\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[x]/(x^K-1) \\ \downarrow \quad \downarrow \\ (\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[G]\times(\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[G]\to(\mathbf{Z}/(2^{2K}+1)\mathbf{Z})[G] \end{array}$$

1.3 高速乗算法の設計法

ショーンハーゲ-ストラッセン法が４つの乗算変換に分解することができるということをみたが、すべての高速乗算法はこのような乗算変換へ分解できることがわかっている。(→ [N5])

高速乗算法が乗算変換へ分解できるという性質を意識することによって、全体の見通しがよくなると同時に、部品となる乗算変換をいくつか用意して、それらをうまく組み合わせることによって高速乗算法を組み立てていくという設計法が可能になる。

ショーンハーゲ-ストラッセン法を変形して、実際に高速乗算法を設計することについて考えてみる。まず多倍長整数を計算機の内部でどのように表現するかであるが、これは現在の計算機の多くが $8 = 2^3$bit の倍数の長さの２進算術演算器をもっているということから、$P = 2^K, K = 2^k (k > 0)$ として P 進表現するとしてよいであろう。

$$a = a_{N-1}P^{N-1} + \cdots + a_1 P + a_0, 0 \leqq a_i < P$$

ションハーゲ-ストラッセン法においては長さ N と K には $N = 2^K$ という関係があったが、これは理論上の計算量を最小にするための工夫であって、このままでは実用には適さない。実際にはこの関係をもっと自由度の高いものにしなければならない。

$$a = a_{N-1}P^{N-1} + \cdots + a_1 P + a_0, 0 \leqq a_i < P$$
$$b = b_{N-1}P^{N-1} + \cdots + b_1 P + b_0, 0 \leqq b_i < P$$

とするとき積 ab を $2^N - 1$ を法として計算すると

$$c = ab \equiv \sum_{r=0}^{N-1} \Big(\sum_{s+t \equiv r (mod N)} a_s b_t \Big) P^r (\mathrm{mod}\ 2^N - 1)$$

となる。この係数 $c_r = \sum_{s+t \equiv r (mod N)} a_s b_t$ の大きさを評価すると、

$$0 \leqq c_r \leqq N(2^K - 1)^2$$

となる。この右側の評価式 $c_r \leqq N(2^K - 1)^2$ を変形して、右辺が孫子の剰余定理でうまく分解できるようにすることが設計の第１段階である。

長さ N の FFT 算法を用意することが設計の第２段階である。この N の選択にあたっては離散フーリエ変換の条件を満たさなければならないという制約がある。この制約は第１段階とも関係し、また 1 の原始 N 乗根の値が計算しやすい形かどうかということも検討すべき項目である。

次節以降、

1. 適当な素数 p を探して、係数の評価式として $c_r < Np$ が成り立つようにする。
2. $N = 2^n, n > 0$ であるとし、FFT 算法として Cooley-Tukey 型算法を使用する。

という方針で高速乗算法を設計し、その Pentium へ実装法について検討する。

2 高速乗算法の設計

2.1 数論変換

我々はすでに可換環上で離散フーリエ変換を定義し、その諸性質をまとめている。(→ [N3]) 工学分野では整数の剰余環上で定義された離散フーリエ変換のことを**数論変換 (NTT, Number Theoretic Transform)** と呼ぶことがある。(→ [S3],[D1])

ここでは法 m と長さ n が与えられたとき、離散フーリエ変換が定義できるための条件について考える。長さの逆元が存在することは $(n,m)=1$ と同値である。法 m が素数のとき、1 の原始 n 乗根の存在については次のことがいえる。

命題 2.1 p が素数のとき、

$$1 \text{ の原始 } n \text{ 乗根} \in \mathbf{Z}/p\mathbf{Z} \Leftrightarrow n|p-1$$

証明 既約剰余類群 $(\mathbf{Z}/p\mathbf{Z})^*$ が位数 $p-1$ の巡回群になることからわかる。■

法 m が合成数のときも既約剰余類群 $(\mathbf{Z}/m\mathbf{Z})^*$ の構造を考えれば、1 の原始 n 乗根の存在する条件がわかる。

離散フーリエ変換の 1 の原始 n 乗根 ζ が存在する場合に条件 3

$$n > k > 0 \rightarrow \sum_{i=0}^{n-1} \zeta^{ik} = 0$$

について考える。

$$(\sum_{i=0}^{n-1} \zeta^{ik})(1-\zeta^k) = 1-\zeta^{kn} = 0$$

が可換環であるという条件だけで成立するが、ここで $1 \neq \zeta^k$ であるから、すべての k について $(1-\zeta^k)$ が単元、すなわち

$$((1-\zeta^k), m) = 1, n > k > 0$$

のとき条件 3 が満たされることがわかる。法 m が素数のとき、この条件はつねに満たされる。

以下、素数 p を法とする場合についてのみ考える。この場合剰余環 $\mathbf{Z}/p\mathbf{Z}$ は有限体 $\mathbf{F}_p$ となる。この単元群 $(\mathbf{F}_p)^*$ は位数 $p-1$ の巡回群であり、その生成元 r は p を法とする**原始根**と呼ばれる。$n|p-1$ なることと 1 の原始 n 乗根 ζ が存在することは同値であり、

$$\zeta = r^{(p-1)/n}$$

と表現される。原始根は一意的には定まらないのでこの表現も一意的ではない。上で示したように $\mathbf{F}_p$ においては 1 の原始 n 乗根 ζ が存在すれば (n,p,ζ) は離散フーリエ変換の条件を満たす。これを定理の形で整理しておく。

定理 2.2 (有限体上の離散フーリエ変換) 素数 p と自然数 n について、

$$n|p-1 \Leftrightarrow \exists \zeta \in \mathbf{F}_p : (n, \mathbf{F}_p, \zeta) \text{ は離散フーリエ変換の型}$$

2.2 数列 T(n)

係数の評価式 $c_r < Np$ における素数 p を探す。

$$c_r \leqq N(2^K-1)^2 = N(2^{2K} - 2^{K+1} + 1)$$

が成り立っているので、

$$2^{2K} - 2^{K+1} + 1 \leqq p$$

なる素数 p であって、$\mathbf{F}_p$ の演算、すなわち法 p の剰余演算が高速に計算できるものを探さなければならない。我々はションハーゲ-ストラッセン法の実装実験 (→ [N4]) において、数論変換を使う高速乗算法は、剰余演算が高速実行できなければ実用にはならないということをすでに経験している。

$$T(n) = 2^{2t} - 2^t + 1, t = 2^n, n > 0$$

とおくと

$$2^{2K} - 2^{K+1} + 1 < T(k), K = 2^k$$

が成り立つ。$T(n), n = 0, 1, 2, 3, 4, 5, 6$ を素数判定してみる。

$$\begin{aligned}
&T(0) = 2(2-1) + 1 = 3 : \text{素数} \\
&T(1) = 2^2(2^2-1) + 1 = 13 : \text{素数} \\
&T(2) = 2^4(2^4-1) + 1 = 241 : \text{素数} \\
&T(3) = 2^8(2^8-1) + 1 = 65281 = 97 * 673 : \text{合成数} \\
&T(4) = 2^{16}(2^{16}-1) + 1 = 4294901761 = 193 * 22253377 : \text{合成数} \\
&T(5) = 2^{32}(2^{32}-1) + 1 = 18446744069414584321 : \text{素数} \\
&T(6) = 2^{64}(2^{64}-1) + 1 : \text{合成数}
\end{aligned}$$

$T(5)$、すなわち $K = 2^5 = 32$ の場合に素数になっているが、これは 32bit アーキテクチャの CPU には大変都合がよい。各係数は

$$A2^{64} + B, 0 \leqq A, B \leqq 2^{64} - 1$$

と表現できるが、このとき、

$$\begin{aligned}
A2^{64} + B &= (2^{64} - 2^{32} + 1)A + (2^{32} - 1)A + B \\
&\equiv (2^{32} - 1)A + B (mod T(5))
\end{aligned}$$

より除算を使わずに剰余演算ができる。(→ [S5])

2.3 離散フーリエ変換の型

$p = T(5) = 2^{32}(2^{32} - 1) + 1$ とする。これは素数であり、

$$c_r < Np$$

という係数の評価式が成り立つ。さらに $\mathbf{F}_p$ 上に長さ N の離散フーリエ変換が存在するための必要十分条件は、

$$N | 2^{32}(2^{32} - 1)$$

であるが、

$$N = 2^n, 0 < n \leqq 32$$

はこの条件を満たし、さらにこのとき p の最小原始根 7 を使って $\mathbf{F}_p$ 上の 1 の原始 N 乗根を

$$7^t, t = 2^{32-n}(2^{32} - 1), 0 < n \leqq 32$$

と表現することができる。

参考文献

計算数論

[N1] 和田 秀男, “コンピュータと素因子分解 改訂版”, 遊星社, 1999

[N2] 和田 秀男, “高速乗算法と素数判定法”, 上智大学数学教室, 1983

[N3] 梅谷 武, 離散 *Fourier* 変換

[N4] 梅谷 武, ストラッセン-ショーンハーゲ法

[N5] Daniel J. Bernstein, *Multidigit multiplication for mathematicians*

算法

[S1] 野下 浩平, 高岡 忠雄, 町田 元, “基本的算法”, 岩波書店, 1983

[S2] D. E. Knuth(中川 圭介訳), “準数値算法/算術演算”, サイエンス社, 1986

[S3] H. J. Nussbaumer(佐川雅彦他訳), “高速フーリエ変換のアルゴリズム”, 科学技術出版社, 1989

[S4] David H. Bailey, *The computation of π to 29,360,000 decimal digits using Borweins' quartically convergent algorithm*

[S5] Mikko Tommila, *Number theoretic transforms*

ディジタル信号処理

[D1] 電子情報通信学会, “ディジタル信号処理の基礎”, コロナ社, 1988

[D2] G. A. Jullien, *Number theoretic techniques in digital signal processing*

[D3] G. A. Jullien, *Residue arithmetic with application in digital signal processing*