数理解析研究所講究録
1198巻 2001年 170-178

多倍長精度の値を係数とする行列の高速乗算方式

日立製作所エンタープライズサーバ事業部 後 保範(Yasunori Ushiro)
Enterprise Server Division, Hitachi Ltd.

1. はじめに

現在の計算機は、10進16桁程度の倍精度演算は高速に行える。また、4倍精度演算もFORTRANやC言語で可能なものも多いが、計算速度は倍精度に比較して1桁程度遅くなる。ここでは、4倍精度以上の多数桁を係数とする連立一次方程式の解の計算における高速化を目的にし、その基本演算となる行列乗算の高速化方式を提案する。

多数桁の乗算には入力値を一定の桁数に分割して定義式で計算する方法以外に中国剰余定理(百五減算)または高速フーリエ変換(FFT)を使用する方式がある。しかし、これらの方式が定義式の計算より高速となるには桁数が数百桁以上と長い必要がある。

一方、多数桁の数値を持つn次元の行列乗算にこれらの方式を適用する場合は、それらの線形性に着目すると中国剰余定理またはFFTの適用回数を$O(n^3)$から$O(n^2)$に削減して、多数桁の乗算が可能になる。今回の提案はそれらの線形性を利用して、多倍長精度を係数にもつn次元の行列乗算に$O(n^2)$回の中国剰余定理またはFFTを適用する高速乗算方式である。中国剰余定理は桁数が数十桁と比較的短いときに有利であり、FFT方式は桁数が十分長い場合に有利となる。

2. 多倍長精度の行列乗算

行列乗算$C=A\cdot B$の計算を多倍長精度で行うものとする。A,B,Cはそれぞれ$n\times l, l\times m, n\times m$次元の行列に適用可能であるが、以下の説明ではすべて$n\times n$次元の行列で、各要素は固定小数点の多倍長精度とする。入力行列A,Bの各要素はm個の数値に分割して保存され、出力行列Cの各要素は入力の2倍強の桁数である$2m+1$個の数値に分割保存されているものとする。また、分割保存された個々の数値の乗算は倍精度浮動小数点演算で正確に実行できるものとする。

$n\times n$次元の行列A,B,Cの各要素を

$$A=(a_{ij}),\ B=(b_{ij}),\ C=(c_{ij}) \qquad i,j=1,2,\cdots,n$$

とし、多数桁の値を持つ入力値の各要素はm個に分割され

$$a_{ij}=a_{ij}^{(1)}\cdot E^{m-1}+a_{ij}^{(2)}\cdot E^{m-2}+\cdots+a_{ij}^{(m-1)}\cdot E+a_{ij}^{(m)}$$

$$b_{ij}=b_{ij}^{(1)}\cdot E^{m-1}+b_{ij}^{(2)}\cdot E^{m-2}+\cdots+b_{ij}^{(m-1)}\cdot E+b_{ij}^{(m)}$$

と表示され、出力値の各要素は$2m+1$個に分割され

$$c_{ij}=c_{ij}^{(1)}\cdot E^{2m}+c_{ij}^{(2)}\cdot E^{2m-1}+\cdots+c_{ij}^{(2m)}\cdot E+c_{ij}^{(2m+1)}$$

と表示されるものとする。

ここで、Eは分割した数値の基本単位で、IEEEの倍精度浮動小数点数では、2進の場合は$E=2^{20}$, 10進の場合は$E=10^6$程度とする。FTTを適用する場合は、丸め誤差を考慮して整数値に正確に戻す必要があるため、$E=2^{13}$ *or* $E=10^4$程度の値とする。

2.1 定義方式

行列乗算$C=A\cdot B$の各要素の計算を下記で行う。

$$c_{ij}=\sum_{l=1}^{n}a_{il}\cdot b_{lj} \qquad i,j=1,2,\cdots,n \tag{2.1}$$

ここで、各要素ごとに多数桁計算を下記で行う。

$$(a\cdot b)^{(k)}=\sum_{p=\max(1,k-m)}^{\min(k,m)}a^{(p)}\cdot b^{(k-p+1)} \qquad k=1,2,\cdots,2m-1 \tag{2.2}$$

(2.1)式を(2.2)式に代入して、行列Cの各要素$c_{ij}(\,i,j=1,2,\cdots,n)$の各分割表示される値$c_{ij}^{(k)}$を下記で計算する。

$$c_{ij}^{(1)}=0,\quad c_{ij}^{(2)}=0$$

$$c_{ij}^{(k+2)}=\sum_{l=1}^{n}\left(\sum_{p=\max(1,k-m)}^{\min(k,m)}a_{il}^{(p)}\cdot b_{ij}^{(k-p+1)}\right) \qquad k=1,2,\cdots,2m-1 \tag{2.3}$$

ここで、$c_{ij}^{(k)}$は

$$\left|c_{ij}^{(k)}\right|\le nm\cdot E^2 \qquad k=3,4,\cdots,2m+1$$

となる。計算された$c_{ij}^{(k)}$は倍精度浮動小数点で正確に表現できる必要がある。

また、Eを単位として、$c_{ij}^{(k)}$は2単位桁上げとなり、$c_{ij}^{(1)},c_{ij}^{(2)}$も桁上げで恒常的な零ではなくなる。さらに、符号は各要素で一括管理し、c_{ij}の成分$c_{ij}^{(k)}$は

$$0\le c_{ij}^{(k)}<E \qquad k=1,2,\cdots,2m+1$$

となるように$c_{ij}^{(k)}/E$の整数部は$c_{ij}^{(k-1)}$に桁上げ処理を行う。

2.2 中国剰余定理を使用し方式

(1) 計算手順

下記のような手順で行列乗算$C=A\cdot B$の各要素$c_{ij},(i,j=1,2,\cdots,n)$を計算する。

(a) 結果c_{ij}が表示可能なようにEに近い素数(正確には互いに素な整数)p_kを$2m+1$個用意する。この時、下記の条件が必要であるが、Eに近い素数を$2m+1$個用意すれば満足する。結果の符号を安全に判定するのにσは4以上が必要。

$$\left|c_{ij}\right|<\left(\prod_{k=1}^{2m+1}p_k\right)/\sigma\approx E^{2m+1}/\sigma \tag{2.4}$$

(b) 入力行列の各要素a_{ij},b_{ij}に対する各素数p_kに対する剰余を計算する。

$$\alpha_{ij}^{(k)}=a_{ij} \qquad \mathrm{mod}(p_k)$$

$$\beta_{ij}^{(k)}=b_{ij} \qquad \mathrm{mod}(p_k) \qquad i,j=1,2,\cdots,n;\quad k=1,2,\cdots,2m+1 \tag{2.5}$$

(c) 各剰余ごとに行列乗算をする。

$$\theta_{ij}^{(k)} = \sum_{l=1}^{n} \alpha_{il}^{(k)} \cdot \beta_{lj}^{(k)} \qquad i,j = 1,2,\cdots,n; \qquad k = 1,2,\cdots,2m+1 \qquad (2.6)$$

(d) 中国剰余定理で結果c_{ij}を計算する

$$c_{ij} = \text{中国剰余定理}(\theta_{ij}^{(1)}, \theta_{ij}^{(2)}, \cdots, \theta_{ij}^{(2m+1)}) \qquad i,j = 1,2,\cdots,n \qquad (2.7)$$

中国剰余定理での計算は下記(2)項で行う。

c_{ij}は下記の形で求める。

$$c_{ij} = c_{ij}^{(1)} \cdot E^{2m} + c_{ij}^{(2)} \cdot E^{2m-1} + \cdots + c_{ij}^{(2m)} \cdot E + c_{ij}^{(2m+1)} \qquad (2.8)$$

但し、$c_{ij}^{(k)}$は下記のようできる。

$$0 \le c_{ij}^{(k)} < E \qquad k = 1,2,\cdots,2m+1$$

(e) 計算したc_{ij}の各分割された値$c_{ij}^{(k)}$を正規化する。

正規化とは符号の決定とそれに伴なう処理である。

中国剰余定理により求めた値は、素数の積（$H = \prod_{k=1}^{2m+1} p_k$）を法とする計算である。

このため、中間結果は正で表現し、本来の結果が負の場合の判定を必要とする。

符号の判定は最上位の桁で行い、定数hを定めておいて、符号は$c_{ij}^{(1)} \le h$なら正とし、それ以外なら符号を負と定め、符号が負の場合c_{ij}は下記のようにする。

$$c_{ij} = H - c_{ij}$$

(2) 中国剰余定理での計算

各$c_{ij}, (i,j = 1,2,\cdots,n)$を$c_{ij} = c_{ij}^{(1)} \cdot E^{2m} + c_{ij}^{(2)} \cdot E^{2m-1} + \cdots + c_{ij}^{(2m)} \cdot E + c_{ij}^{(2m+1)}$

の形で表示し、以下の計算を行う。

(a) 前処理

$$c_{ij} = c_{ij}^{(2m+1)} = \theta_{ij}^{(1)}$$

(b) 主計算

下記を$k = 2$から$2m+1$まで繰り返す。

$$v = c_{ij} \qquad \mathrm{mod}(p_k)$$

$$w = (\theta_{ij}^{(k)} - v) \cdot t_k \qquad \mathrm{mod}(p_k)$$

$$c_{ij} = c_{ij} + w \cdot Q_k \qquad (2.9)$$

但し、多倍長精度の定数Q_kと定数t_kは前もって求めておく。

$$Q_k = \prod_{l=1}^{k-1} p_l$$

$$t_k = 1/Q_k \qquad \mathrm{mod}(p_k) \qquad k = 2,3,\cdots,2m+1$$

(3) 計算のための注意事項

(a) $\alpha_{ij}^{(k)} = a_{ij} \quad \mathrm{mod}(p_k)$等の剰余の計算

$d_l^{(k)} = E^{(l)} \ \mathrm{mod}(p_k)$を先に計算しておき、$a_{ij}$が$a_{ij}^{(k)}$で表示されるのを利用して次のように計算する。

$$\alpha_{ij}^{(k)} = \sum_{l=0}^{m-1} a_{ij}^{(m-l)} \cdot d_l^{(k)} \qquad \mathrm{mod}(p_k) \qquad k = 1,2,\cdots,2m+1;\ i,j = 1,2,\cdots,n$$

(b) 符号の決定に使用する基準値hの算出

$H = \prod_{k=1}^{2m+1} p_k$ を計算し、下記のように表示する。

$$H = \prod_{k=1}^{2m+1} p_k = h^{(1)} \cdot E^{2m} + h^{(2)} \cdot E^{2m-1} + \cdots + h^{(2m)} \cdot E + h^{(2m+1)} \qquad (2.10)$$

一方、(2.4) 式から

$$-H/\sigma < c_{ij} < H/\sigma \qquad \sigma \geq 4$$

となる。このため

$$h = h^{(1)}/2$$

としておけば、(2.9)式で求めた$c_{ij}^{(1)}$とhを比較して、c_{ij}の正負が判定できる。

2.3 FFT を使用した方式

(1) 計算手順

下記のような手順で行列乗算$C = A \cdot B$の各要素$c_{ij}, (i,j = 1,2,\cdots,n)$を計算する。

FFT 変換には、実 FFT 変換と素数Pを法とする整数 FFT 変換[2]があるが、現在の計算機での効率を考慮して実 FFT 変換での方式を採用する。

(a) 入力値a_{ij}, b_{ij}に対して順実 FFT 変換をする。

下記のように表現されている入力値a_{ij}, b_{ij}に対して

$$a_{ij} = a_{ij}^{(1)} \cdot E^{m-1} + a_{ij}^{(2)} \cdot E^{m-2} + \cdots + a_{ij}^{(m-1)} \cdot E + a_{ij}^{(m)}$$

$$b_{ij} = b_{ij}^{(1)} \cdot E^{m-1} + b_{ij}^{(2)} \cdot E^{m-2} + \cdots + b_{ij}^{(m-1)} \cdot E + b_{ij}^{(m)}$$

後半の要素をゼロにした、それぞれ$2m$個の要素を持つ下記のようなベクトル$\tilde{a}_{ij}, \tilde{b}_{ij}$を作成する。

$$\tilde{a}_{ij} = (a_{ij}^{(1)}, a_{ij}^{(2)}, \cdots, a_{ij}^{(m)}, 0, \cdots, 0)$$

$$\tilde{b}_{ij} = (b_{ij}^{(1)}, b_{ij}^{(2)}, \cdots, b_{ij}^{(m)}, 0, \cdots, 0) \qquad i,j = 1,2,\cdots,n$$

$\tilde{a}_{ij}, \tilde{b}_{ij}$に対して順実 FFT 変換を行う。

$$\bar{a}_{ij} = FFT(\tilde{a}_{ij})$$

$$\bar{b}_{ij} = FFT(\tilde{b}_{ij}) \qquad i,j = 1,2,\cdots,n \qquad (2.11)$$

(b) a_{ij}, b_{ij}の FFT 変換結果に対して項別に内積計算する。

$$f_{ij} = \sum_{l=1}^{n} \bar{a}_{il} \otimes \bar{b}_{lj} \qquad i,j = 1,2,\cdots,n \qquad (2.12)$$

ここで、⊗はベクトルの対応する要素ごとの計算である。

$\bar{a}_{ij}, \bar{b}_{ij}, f_{ij}$を下記のように表現すると

$$\bar{a}_{ij} = (\bar{a}_{ij}^{(1)}, \bar{a}_{ij}^{(2)}, \cdots, \bar{a}_{ij}^{(2m)})$$

$$\bar{b}_{ij} = (\bar{b}_{ij}^{(1)}, \bar{b}_{ij}^{(2)}, \cdots, \bar{b}_{ij}^{(2m)}) \qquad i,j = 1,2,\cdots,n$$

$$f_{ij}=(f_{ij}^{(1)},f_{ij}^{(2)},\cdots,f_{ij}^{(2m)})$$

具体的計算は次のようになる。

$$f_{ij}^{(1)}=\sum_{l=1}^{n}a_{il}^{(1)}\cdot b_{lj}^{(1)} \qquad f_{ij}^{(m+1)}=\sum_{l=1}^{n}a_{il}^{(m+1)}\cdot b_{lj}^{(m+1)}$$

$$f_{ij}^{(k)}=\sum_{l=1}^{n}\left(a_{il}^{(k)}\cdot b_{lj}^{(k)}-a_{il}^{(k+m)}\cdot b_{lj}^{(k+m)}\right) \qquad i,j=1,2,\cdots,n;\quad k=2,3,\cdots,m$$

$$f_{ij}^{(k+m)}=\sum_{l=1}^{n}\left(a_{il}^{(k)}\cdot b_{lj}^{(k+m)}+a_{il}^{(k+m)}\cdot b_{lj}^{(k)}\right) \qquad (2.13)$$

(c) 未正規化結果$\hat{c}_{ij}$を逆実FFT変換により求める。

$$\hat{c}_{ij}=FFT^{-1}(f_{ij}) \qquad i,j=1,2,\cdots,n$$

(d) $\hat{c}_{ij}$の各分割された値$\hat{c}_{ij}^{(k)}$を正規化し結果c_{ij}を求める。

正規化とはFFT結果を整数値に戻すこと、c_{ij}の符号の確定及び桁上げである。

各$\hat{c}_{ij},(i,j=1,2,\cdots,n)$を

$$\hat{c}_{ij}=\hat{c}_{ij}^{(1)}\cdot E^{2m-1}+\hat{c}_{ij}^{(2)}\cdot E^{2m-2}+\cdots+\hat{c}_{ij}^{(2m-1)}\cdot E+\hat{c}_{ij}^{(2m)}$$

と表示したとき、整数値に戻すのは、各$\hat{c}_{ij}^{(k)}$が誤差のため、実数となっているのを、本来は整数値であるため、その実数に最も近い整数値に戻す処理である。

c_{ij}の符号の確定は$c_{ij}^{(k)}\geq 0$となるように、多数桁c_{ij}の正負を定める。

桁上げ処理は、

$$-nm\cdot E^2<\hat{c}_{ij}^{(k)}<nm\cdot E^2$$

となっている、$\hat{c}_{ij}^{(k)}$にたいしEを超えた値を桁上げして、$c_{ij}^{(k)}$を下記の範囲に収める処理である。この時、$\hat{c}_{ij}^{(k)}$を$c_{ij}^{(k+1)}$に対応させる。

$$0\leq c_{ij}^{(k)}<E \qquad k=1,2\cdots,2m+1$$

(2) 計算のための注意事項

実FFTを使用して、多倍長精度の乗算を行うため、整数値を入力しても、乗算結果は誤差を含むため整数値とならない。そのため

$$\hat{c}_{ij}=FFT^{-1}(f_{ij}) \qquad i,j=1,2,\cdots,n$$

で計算した、各成分$\hat{c}_{ij}^{(k)},(k=1,2,\cdots,2m)$は小数以下の値を持つ実数となっている。

$\hat{c}_{ij}^{(k)}$を整数化するには、$\hat{c}_{ij}^{(k)}$に一番近い整数を採用すれば良いが、その時整数化誤差(ε)を下記のように求める。

$$\varepsilon=\max_{k,i,j}\left(\left|\text{整数化}\hat{c}_{ij}^{(k)}-\hat{c}_{ij}^{(k)}\right|\right) \qquad k=1,2,\cdots,2m;\quad i,j=1,2,\cdots,n$$

一般に、$\varepsilon<0.2$なら、正確に整数化される。

3. 各方式による計算量

3.1 演算量算出のための計算条件

$n\times n$次元行列で、入力はmワード、結果は$2m+1$ワードで表現される多倍長精度の

整数値とする。ワード単位に分割した値は、倍精度浮動小数点形式で保存し、分割した値どうしの乗算及び内積を正確に表現可能な桁数とする。また、それぞれの演算量は下記の通りとする。定義方式及び中国剰余定理方式では、1ワード(倍精度浮動小数点)に10進6桁つめ($E=10^6$)とし、FFT方式では誤差を考慮して1ワードに10進4桁つめ($E=10^4$)とする。

(a) 分割した値どうしの乗算： 1

(b) n個の要素の内積演算 ： $2n$

(c) $\mathrm{mod}(p_k)$等の剰余計算 ： 4

(d) FFT計算

計算要素数(誤差を考慮し)： $3m$

実FFT変換の演算量 ： $9m\cdot\log_2(3m)$

(e) 計算結果の符号の判定や、多倍長精度への正規化の演算量は比較的少ないので考慮しない。

3.2 定義方式の計算量

mワードに分割した、多倍長精度の乗算がn^3回必要である。1回の多倍長精度の演算量は$2m^2$である。桁上げのために$2mn^2$回の$\mathrm{mod}(E)$が必要であるが、この部分は無視すると定義方式の演算量T_0は次のようになる。

$$T_0 = 2m^2n^3$$

3.3 中国剰余定理での演算量

中国剰余定理での演算量を下記に示す。

(a) a_{ij}, b_{ij} $\mathrm{mod}(p_k)$の計算

演算量及び回数の内訳は下記のようになる。

多数桁要素1個の$\mathrm{mod}(p_k)$の演算量： $2m$

a_{ij}, b_{ij}の個数 ： $2n^2$

p_kの個数 ： $2m+1$

従って、a_{ij}, b_{ij} $\mathrm{mod}(p_k)$の演算量T_{11}は次のようになる。

$$T_{11} = 4m(2m+1)n^2$$

(b) (2.6)式での$\theta_{ij}^{(k)} = \sum_{l=1}^{n} \alpha_{il}^{(k)} \cdot \beta_{lj}^{(k)}$の計算

演算量及び回数の内訳は下記のようになる。

1回の内積計算の演算量 ： $2n$

$\theta_{ij}^{(k)}$のi, jの回数(行列の要素数) ： n^2

$\theta_{ij}^{(k)}$のkの数(素数p_kの個数) ： $2m+1$

従って、$\theta_{ij}^{(k)}$の演算量T_{12}は次のようになる。

$$T_{12} = 2(2m+1)n^3$$

(c) c_{ij} =**中国剰余定理**$(\theta_{ij}^{(1)},\theta_{ij}^{(2)},\cdots,\theta_{ij}^{(2m+1)})$ の計算

演算量及び回数の内訳は下記のようになる。

中国剰余定理を 1 回適用する演算量は(2.11)式のvとc_{ij} の計算が主力となり、

それぞれの演算量は

$$\sum_{l=1}^{2m} 2l = 2m(2m+1) \text{ 及び } \sum_{l=1}^{2m} 6l = 6m(2m+1)$$

となる。合計で$8m(2m+1)$ となる。

中国剰余定理の適用回数：n^2

従って、中国剰余定理によるc_{ij} の計算の演算量T_{13} は次のようになる。

$$T_{13} = 8m(2m+1)n^2$$

(d) 合計演算量

合計演算量T_1 はT_{11},T_{12},T_{13} の合計で次のようになる。

$$T_1 = 4m(2m+1)n^2 + 2(2m+1)n^3 + 8m(2m+1)n^2$$
$$= 2(2m+1)(6m+n)n^2$$

3.4 FFT での演算量

FFT での演算量を下記に示す。FFT を使用した計算では誤差を考慮するため定義式や中国剰余定理の場合と異なり、1 ワードに 10 進 6 桁ではなく 10 進 4 桁つめとして演算量を計算する。そのため、計算手順で示すm は 1.5 倍し、FFT の要素数は$3m$ となる。

(a) FFT 変換

演算量及び回数の内訳は下記のようになる。

1 回の FFT 変換の演算量 ： $9m\cdot\log_2(3m)$

FFT 変換の回数 ： $3n^2$

従って、順 FFT 変換の演算量T_{21} は次のようになる。

$$T_{21} = 27mn^2\cdot\log_2(3m)$$

(b) FFT 変換結果の項別内積計算

項別内積計算の演算量T_{22} は次のようになる。

$$T_{22} = 12mn^3$$

(c) 合計演算量

合計演算量T_2 はT_{21},T_{22} の合計で次のようになる。

$$T_2 = 27mn^2\cdot\log_2(3m) + 12mn^3 = 3mn^2(9\log_2(3m) + 4n)$$

4. 適用結果

多倍長精度を係数とする行列乗算$C = A\cdot B$ の演算量及び計算時間の評価を行う。提案方式を評価するために、$n\times n$ 次元の行列で 1 ワードに 10 進 6 桁とし、入力はm 倍精度で出力は$2m+1$ 倍精度としたときの演算量及び計算時間の比較評価を行う。

FFT 方式の場合は 1 ワードに 10 進 4 桁つめで計算するが、演算量及び計算時間の比較評価では、1 ワードに 10 進 6 桁つめにした場合に換算して評価する。

性能評価には下記の計算機とコンパイラを使用した。

(a) 使用計算機： Pentium 200Mhz

(b) 使用言語 ： FORTRAN

(c) コンパイラ： DIGITAL Visual FORTRAN V6.0A (Full Optimization 指定)

図 1、図 2、図 3 に浮動小数点倍精度演算を使用した同一次元の行列乗算における計算時間を 1 にした比較結果を示す。図 1 は 50 次元の行列で、図 2、図 3 は 100 次元及び 200 次元に対する m 倍精度を入力値とする計算時間比を示したものである。

図 4 に 3 章で算出した演算量をもとに行列乗算の演算量の比較を示す。比較のもとは図 1,2,3 と同じく浮動小数点倍精度演算を使用した行列乗算の演算量を 1 にした比較である。

図 5、図 6 に定義方式の計算時間を 1 にした場合の提案方式の効果を示した。図 5 は 50,100 及び 200 次元における中国剰余定理の効果である。図 6 は 50,100 及び 200 次元における FFT 方式の効果である。

これらの図から、中国剰余定理を使用した方式は 4 倍精度で既に、定義方式より高速で、通常の浮動小数点倍精度演算での行列乗算に対し約 10 倍の時間で実行できることが分かる。また、8,12 及び 16 倍精度での計算は 4 倍精度のそれぞれ 2 倍, 3 倍及び 4 倍で実行でき、桁数が短いうちはほぼ桁数に比例した時間となることが分かる。

中国剰余定理は FFT 方式に比較し、入力値の精度が比較的小さいとき有利で、その分かれ目は行列の次元数に比例して大きくなる。

図 1,2,3,4 の定義は定義方式、剰余は中国剰余定理(百五減算)、FFT は FFT 方式を多倍長精度の値を係数とする行列乗算に適用することを意味する。図 5,6 の $n=50$、$n=100$ 及び $n=200$ はそれぞれ 100 次元、200 次元及び 300 次元の行列乗算を示す。

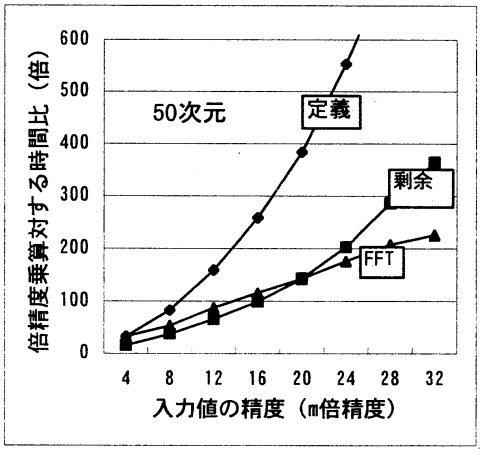

図 1 実行時間の比較(50 次元)

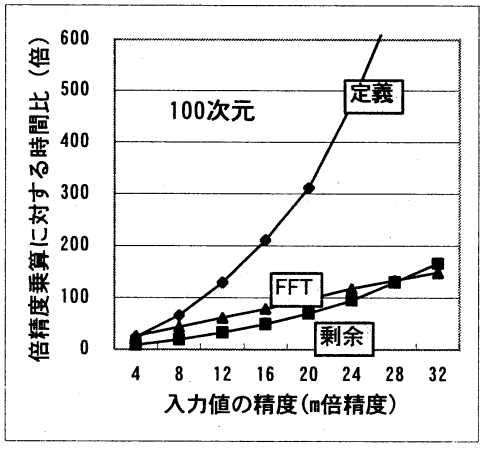

図 2 実行時間の比較(100 次元)

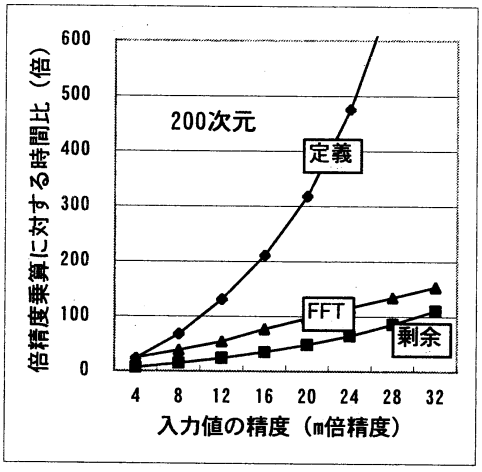

図3 実行時間の比較(200次元)

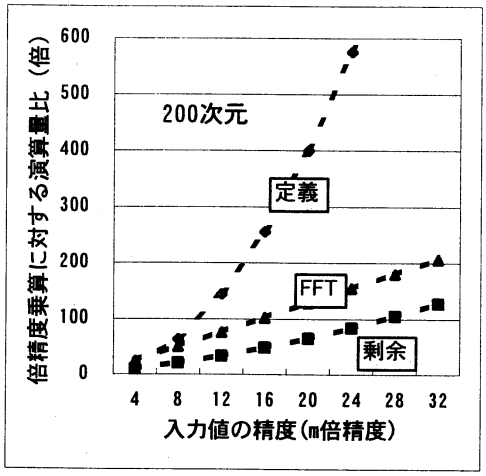

図4 演算量の比較(200次元)

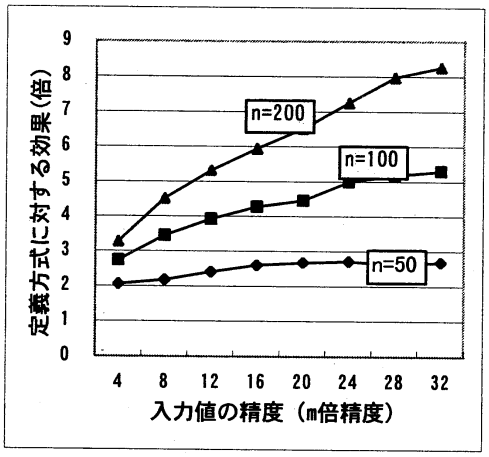

図5 定義方式と中国剰余定理の比較

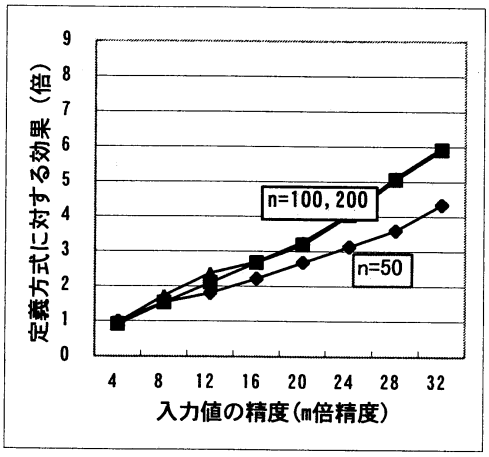

図6 定義方式とFFT方式の比較

5. まとめ

多倍長精度を係数とする行列乗算に対して、2方式の高速化を提案した。一つは中国剰余定理を利用する方法であり、もう一つはFFTを利用する方式である。

中国剰余定理を利用する方法は、入力値が10進6桁を単位とする4倍精度で既に、定義方式より高速となり、比較的桁数が短いときに効果を発揮することを示した。一方、FFT方式は桁数が大きな行列乗算に有効な方式である。

今後は、多倍長精度で解を求める必要がある連立一次方程式の直接解法及び反復解法への応用を検討する。

6. 参考文献

1) 後 保範： 多倍長精度の値を係数とする行列の高速乗算方式、京大数理研予稿集 [偏微分方程式の数値解法とその周辺2], Nov 2000.

2) 後 保範：ベクトル計算機による整数上のFFT計算、情報処理全国大会、1983後期