

ICCG 法の並列計算

坪井 始* 瀬島紀夫** 田中始男*

Parallel Computation of the ICCG Method

Hajime TSUBOI* Norio SESHIMA** Motoo TANAKA*

ABSTRACT

Various types of numerical analysis methods for electromagnetic problems have been established and finite element methods and integral equation methods are practically used in the design of electrical machines. On the other hand, the technology of personal computer hardware is rapidly progressing and practical electromagnetic computation can be performed by a personal computer. Recently high end machines are equipped with multi-CPU and multi-core. This means that parallel computation can be done by the personal computer without special hardware. In this paper, parallelization of the ICCG method is investigated. The ICCG method is a suitable solver for finite element methods using edge elements but parallelization of the method is difficult because of dependent data structure. Effective parallelization principles are described using some simple examples and effectiveness of the parallelized ICCG method is shown.

Keywords: parallelization, Open MP, SMP, ICCG method, FEM, electromagnetic field computation.

1. まえがき

電磁界の数値解析は、有限要素法、積分方程式法等、様々な手法が確立され、実用化されている。一方、コンピュータハードウェア技術の進歩は目覚ましく、パーソナルコンピュータでも十分実用的な解析ができるようになってきた。最近では、ハイエンド・パーソナルコンピュータは、マルチCPU化、マルチコア化が進んでいる。これは、特別なハードウェアの追加なしに並列計算が可能になることを意味し、共有メモリ型の並列計算が容易に行えるようになる。

本稿では、有限要素法による電磁界解析プログラムの中で、連立一次方程式法の解法であるICCG(Incomplete Cholesky Conjugate Gradient Method)法の並列化について検討する。

分割データの入力、連立一次方程式の係数行列の作成、連立一次方程式の求解、計算結果の出力の4つのプロセスに大きく分けることができる。並列処理が容易で、その効果が最も期待できるのは、使用データに依存関係のない係数行列の作成であり、ほぼリニアな並列効果が得られる。一方、問題によっては計算時間がかかり、並列化が難しいのは、連立一次方程式の解法であるICCG法である。

ICCG法は、不完全コレスキー分解による前処理付き共役勾配法で、電磁界解析で用いられる辺要素有限要素法に適した連立一次方程式の解法である。通常、電磁界解析で使用されるICCG法は、オリジナルな解法からいくつかの改良がなされている。係数行列や未知変数の複素数化、係数行列のスケーリング、加速係数の導入[1]が、その改良である。

2. 有限要素法プログラムとICCG法

有限要素法による電磁界解析プログラムは、有限要素

3. 使用した計算環境

本稿の計算に使用したハードウェアおよびソフトウェア

* 情報処理工学科 ** 情報処理センター

表1 ハードウェア仕様

Table 1 Hardware.

CPU (64 bit)	Power PC 970MP 2.5GHz x2 (G5 Dual CPU, Quad Core)
二次キャッシュ	1 MB/Core
フロントサイドバス	1.25 GHz/CPU
メモリ	8 GB (PC2-4200)

表2 ソフトウェア仕様

Table 2 Software.

OS	Mac OS X 10.4.8 (Darwin 8.8.0)
コンパイラ	IBM XL Fortran Advanced Edition for Mac OS X ver. 8.1
コンパイルコマンドとオプション	
逐次処理用	xlf -O3
並列処理用	xlf_r -O3 -qsmp=omp

アの仕様を表1および表2に示す。CPUは、64ビットで、Dual Core仕様のものをDual CPUとして使用しているため、Quad Core（4つのプロセッサ使用可能）となり、4つのスレッドを同時に並列処理可能である。オペレーティングシステムは、UNIXシステムの1つであるMac OS Xである。コンパイラは、Power PC専用のFortranコンパイラで、逐次処理の場合と並列処理の場合では、異なるコンパイラモジュールを使用するようになっている。

4. Open MPによる並列処理

並列処理には、Open MPを用いた[2]。Open MPは、共有メモリシステムを用い、スレッドの並列化によって並列処理を行う仕様である。共有メモリシステムの中で、各プロセッサが同等の条件でメモリ空間にアクセス可能なものはSMP (Symmetric Multiprocessor) と呼ばれるが、本稿で使用したハードウェアもSMPシステムである。また、Open MPは、ライブラリではなく、逐次計算プログラムに指示文を挿入していくことにより、並列化を行う。この指示文は、Open MPをサポートしないコンパイラでは単なるコメントとみなされ、無視されるので、逐次計算プログラムとして動作保証されるとい

う、特徴がある。本稿では、Fortran77で作成されたプログラムをOpen MPを用いて並列化した。

4.1 簡単な並列処理

まず、次の数列の和を求める並列化し、スレッド生成のオーバーヘッドを確認した。

$$S = \sum_{i=1}^N i^2 = \frac{N(N+1)(2N+1)}{6} \quad (1)$$

リスト1 数列の和の計算プログラム

List 1 Program for the sum of a progression.

```

implicit double precision (a-h,o-w)
N=2**20
write(*,*) 'Calc Sum(n*n), n=',N
s = N
L = ( 2**30 / N ) * 10

do 10 k=1,L
  c = 0d0

!$omp parallel do private(a), reduction(+:c)
  do 20 i=1,N
    a = i
    c = c + a * a
20  continue

10 continue
write(*,*) 'C      =', c
end
    
```

number of threads

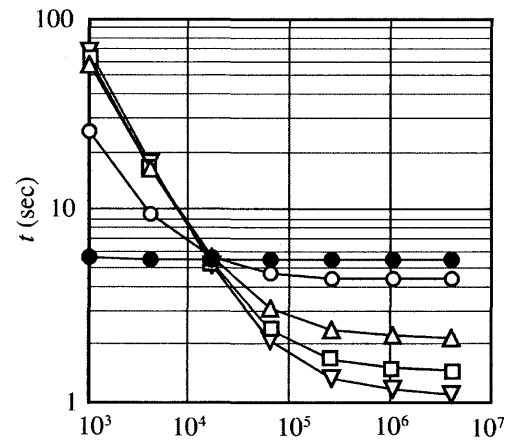
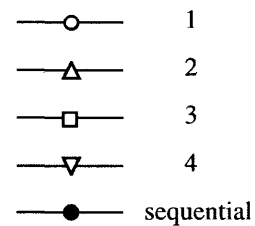


図1 数列の和の計算時間

Fig. 1 Computation time of the sum of a progression.

リスト1に、そのプログラムを示す。!\$omp で始まる文が、Open MPの指示文である。Nは、並列計算を行うdoループの長さとなり、それをL回繰り返すことによって、NxLを一定として同じ回数積和を計算するプログラムである。したがって、Nを小さくするとスレッドの発生回数が多くなるが、積和の計算量は同じである。

リスト1のプログラムを用いて、Nを変化させたときの計算時間tを図1に示す。プログラムのコンパイルは、逐次処理用にはコマンドxlfを、並列処理用にはコマンドxlf_rを使用している。並列処理のスレッド数は、環境変数によって指定し、スレッド数を、1、2、3、4としている。逐次処理の計算時間は、積和の計算量が一定であるので、ほぼ一定となっている。また、スレッド数が1の場合もスレッドを生成しているため、逐次処理とは計算時間が異なり、スレッドの生成回数が多いときにはそのオーバーヘッドで計算時間が長くなり、少ないときには一定となるが、処理プログラムが異なるので計算時間は逐次処理よりも短くなっている。これらの計算結果より、スレッド生成に関するオーバーヘッドは数μsecから10μsec、1回の積和計算時間は約0.5nsecと見積もることができる。この結果と図1より、積和計算では10万回以上のループ長で並列処理の効果が現れることがわかる。すなわち、並列計算の効果を上げるには、ループの1回当たりの計算量から決まるあるループ長以上の長さのループを作成する必要があることが確認できた。したがって、1回のループ内の計算量が多ければ短いループ長でも並列化の効果が上げられることになる。

4.2 ガウスの消去法の並列処理

つぎに、ガウスの消去法のプログラムの並列化を行った。リスト2にプログラムを示す。並列化には、ホットスポットと呼ばれる計算量が多く、計算時間のかかる部分を特定し、これを並列化するのが最も効果的であるが、計算結果が計算順序に依存しないことが必要である。ここでは、前進消去のピボット行による消去部分(リスト中では制御変数kのループ)が並列化可能である。この部分は、前進消去が進むと計算量が減少して並列効果も減少することが考えられるが、特に考慮していない。また、後退代入部分は、逐次的に処理する必要があるため並列化には適さず、計算量も前進消去に比べてわずかであるので、並列化していない。

表3に、リスト2のプログラムによるガウスの消去法の計算時間を示す。コンパイルには、並列処理用コマンドを使用している。同表は、プログラム中のNSIZEを6000とし、5000元の連立一次方程式を解いた場合の計算時間である。スレッド数を2とした場合に、スレッド数1の場合と比べて計算時間が75.6%に短縮されているが、スレッド数を3、4と増やしても並列化の効果

はほとんど変化がない。これは、プログラムの並列化部分が、計算が単純でメモリアクセスが非常に多いプログラムとなっているためと考えられる。すなわち、並列化部分で、メモリアクセスの競合がおき、並列化の効果が上がらないものと考えられる。使用したシステムでは、メモリアクセスのためのフロントサイドバスは、CPU毎に1系統、合計2系統であるため、スレッド数2で並列化の効果が飽和すると考えられる。ガウスの消去法の場合、逐次処理のプログラムのベンチマークテスト[3]

リスト2 ガウスの消去法プログラム
List 2 Program of the Gauss elimination method.

```

subroutine gauss(n,cc,NSIZE)
implicit double precision (a-h,o-w)
dimension cc(NSIZE+1,NSIZE)
np=n+1

do 1000 k=1,n-1
!$omp parallel do private(ci,j)
do 10 i=k+1,n
ci=-cc(k,i)/cc(k,k)
do 50 j=k+1,np
cc(j,i)=cc(j,i)+cc(j,k)*ci
50 continue
10 continue
1000 continue

cc(np,n)=cc(np,n)/cc(n,n)
do 2000 i=n-1,1,-1
do 30 j=n+1,n
cc(npp,i)=
& cc(npp,i)-cc(npp,j)*cc(j,i)
30 continue
cc(np,i)=cc(np,i)/cc(i,i)
2000 continue

return
end

```

表3 ガウスの消去法の並列計算
Table 3 Parallel computation of the Gauss Elimination method.

Number of threads	Computation time (sec)	ratio (%)
1	156.	100.0
2	118.	75.6
3	116.	74.3
4	116.	74.3

を行った場合も、メモリのアクセス速度の影響が大きいことが知られており、このことからメモリアクセスの競合が並列化の効果に大きく影響することが類推できる。また、前述したように、並列化部分の計算が進むにつれて計算量が減少し、前節で述べたスレッド生成のオーバーヘッドによって並列化の効果が上がらないことも考えられる。リスト2のプログラムでは、並列化の効果が上がらない理由はいくつか考えられ、チューニングの方法もあるが、ここでは、メモリアクセスの競合が並列化効果へ大きく影響することの確認にとどめる。

5. ICCG法の並列計算

ICCG法[4]は、対称疎行列の連立一次方程式を解く手法で、三次元電磁界の有限要素解析には不可欠な連立一次方程式の求解法である。ICCG法は、反復法として収束性がよく、前処理の後CG法を用いるので、反復回数は元数nのオーダーである。電磁界解析に用いた場合には、nの数%、場合によっては1%以下の反復回数で収束する。必要とするメモリ容量も、基本的に非零要素のみを格納すればよいので、比較的小さい。電磁界解析用には、オリジナルのICCG法から次のような改良がなされている。電磁界の線形交流解析では、複素数の係数行列および未知変数が用いられるので複素数に拡張されている。実数用のICCG法はほとんどプログラムを変更しないで複素化可能である。つぎに、材料定数の大きな違いから係数行列の対角要素の絶対値に大きな差が生じるので収束性の改善のため、通常スケールリングを行う。さらに、収束性の改善のため不完全コレスキー分解の際に、対角項に加速係数を導入する[1]。

解くべきn元連立一次方程式を次式で与える。

$$A_{org} X_{org} = b_{org} \quad (1)$$

ここで、 A_{org} は係数行列、 X_{org} は未知変数、 b_{org} は定数ベクトルである。

有限要素法の係数行列は対称行列であるので、ICCG法のプログラムには、右上三角行列中の非零要素と対角要素、定数ベクトルのみを入力する。このとき、非零要素と同時にその列番号を整数配列に格納する。

ICCG法の計算手順はつぎの通りである。

(I) スケーリング

$$SA_{org} S^{-1} X_{org} = S b_{org} \quad (2)$$

$$A = SA_{org} S^{-1}$$

$$X = S^{-1} X_{org}$$

$$b = S b_{org}$$

ここで、

$$s_{ii} = 1/a_{org\ ii}^{1/2}$$

$$s_{ij} = 0 \quad (i \neq j)$$

解くべき方程式は、つぎのようになる。

$$AX = b \quad (3)$$

(II) 右上三角行列を転置し、左下三角行列を作成

(III) 不完全コレスキー分解 (L と D の計算)

$$A \approx LDL^T \quad (4)$$

ここで、初期値 $l_{ij} = a_{ij}$ として、

$$l_{ij} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} d_k \quad (i \neq j)$$

$$l_{ii} = \gamma a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2 d_k$$

$$d_i = 1/l_{ii}$$

である。 γ は、加速係数。

(IV) CG法

(i) 初期化

$$r_0 = b - AX_0 \quad (X_0 = 0)$$

$$P_0 = (LDL^T)^{-1} r_0$$

(ii) 反復計算

$$\alpha_k = \frac{(r_k \cdot (LDL^T)^{-1} r_k)}{(P_k \cdot AP_k)} = \frac{(r_k \cdot U_k)}{(P_k \cdot AP_k)} \quad (5)$$

$$U_k = (LDL^T)^{-1} r_k$$

$$X_{k+1} = X_k + \alpha_k P_k \quad (6)$$

$$r_{k+1} = r_k - \alpha_k AP_k \quad (7)$$

$$\|r_{k+1}\| / \|b\| \leq \varepsilon \quad (8)$$

$$\beta_k = \frac{(r_{k+1} \cdot U_{k+1})}{(r_k \cdot U_k)} \quad (9)$$

$$P_{k+1} = U_{k+1} + \beta_k P_k \quad (10)$$

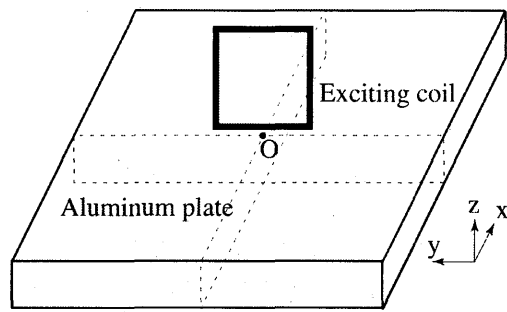
(8) 式の ε は、収束判定値で、同式を満たした場合に反復計算を終了する。

(V) 未知変数を求める

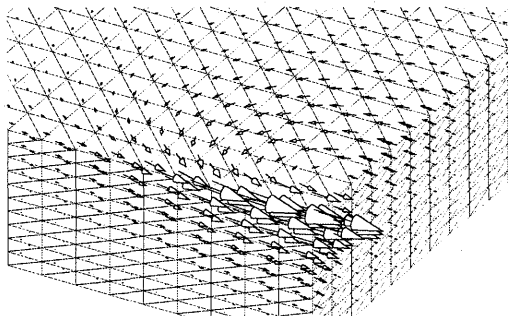
$$X_{org} = SX \quad (11)$$

不完全コレスキー分解における加速係数 γ は、1 から 1.5 の値をとるが、有限要素分割や材料定数の値によって係数行列の性質が変化し、その最適値は異なる。加速係数の効果は非常に大きく、その値の決定には注意を要する。

図2の渦電流探傷の計算モデルを用いてICCG法の計算時間および並列処理の効果を検討した[5-7]。同図(a)に示すようにアルミニウム板の上に矩形コイルが垂直に置かれたモデルである。同図(b)は、対称性を考慮して解析領域を1/4にしたときの、アルミニウム板に



(a) computation model



(b) eddy current distribution

図2 渦電流探傷の計算モデル

Fig. 2 Computation model for eddy current testing.

誘導される渦電流を示す。この場合は、四面体要素数が66,150、複素数の未知変数が105,485であり、係数行列の収束特性が良いモデルである。右上三角行列の1行当たりの非零要素の最大数は23であり、加速係数 γ を1.1、収束判定値 ε を 10^{-8} として計算した場合の収束までの反復回数は551である。

表4に、ICCG法プログラム内各プロセスの計算時間を示す。CG法の反復1回当たりの計算時間と比べると、スケーリングの計算時間は短く、入力行列の転置と不完全コレスキー分解の計算時間は同程度か、長い。しかし、CG法は551回反復されるので、全体の計算時間では、CG法の計算時間が支配的である。一方、並列化の観点からは、スケーリングと転置は、並列化が容易であるが、全体の計算時間に対してはほとんど計算時間の短縮効果がない。

CG法の中では、 AP_k の計算は計算量があり、並列化の効果が大きい。また、(6)~(8)式の計算を同一のdoループで計算し、並列化している。他にも、doループで並列化可能と考えられるものがあるが、ループ長と計算量を考慮して上記二箇所の並列化を行った。これらの並列化によって表4のような効果が得られた。ICCG法全体の計算時間は、表5のようになり、並列化の効果は十分とは言えない。

スレッド数の増加によって計算時間が短縮しているこ

表4 ICCG法の計算時間

Table 4 Computation time of the ICCG method.

Scaling coefficient matrix	0.04 sec
Transpose input matrix	0.15 sec
IC decomposition	0.51 sec
CG method	
number of threads: 1	0.142 sec/iteration
2	0.120
3	0.113
4	0.110

表5 ICCG法の並列計算

Table 5 Parallel computation of the ICCG method.

Number of threads	Computation time (sec)	ratio (%)
1	78.8	100.
2	66.9	84.9
3	62.9	79.8
4	61.0	77.4

とから、並列化の効果はあるが、全計算時間に占める割合が小さい。CG法の反復計算の中では、 U_k の計算時間が長い。計算が逐次的で並列化に適さない。さらに、並列化の効果の向上、計算時間の短縮には、 U_k の計算法の検討、配列の使用法、doループの書き方等、詳細な検討が必要である。

6. むすび

有限要素法による電磁界解析で不可欠なICCG法のOpen MPを用いた並列計算について検討した。その結果、つぎのことを確認した。

(1) doループの並列処理では、スレッド生成のオーバーヘッドを考慮する必要がある。計算量の少ないdoループでは、並列化により計算時間が増加することがある。

(2) メモリアクセスの多い計算プログラムの並列処理では、メモリアクセスの競合が起り、期待する並列効果が得られないことがある。

従来、指摘されているように、プログラムの中で並列化の効果を上げるためには計算量が多く、全計算時間に

占める割合の多い箇所を特定し、並列化を行う必要があるが、上記の事柄にも注意を払う必要がある。また、上記二点は、使用する計算機システムのアーキテクチャや使用するパーツ、オペレーティングシステムの機能、コンパイラの性能に影響されるため、少なからずチューニングが必要である。最近では、コンパイラの自動並列化機能も性能が向上し、簡単なプログラムでは良好な結果が得られる場合があるが、複雑なプログラムでは十分な性能が得られていない。また、並列化に適したアルゴリズムの開発は重要で、並列化効果を上げるための最も重要なファクターである。

参考文献

- [1] 藤原、中田、房安：ICCG 法の高速度化手法に関する検討、電気学会静止器・回転機合同研究会資料、SA-91-43・RM-91-106、1991.
- [2] 牛島：Open MP による並列プログラミングと数値計算法、p. 11、2006、丸善。
- [3] 坪井、内藤編著：実践電磁界解析法、p. 6、1995、養賢堂。
- [4] 戸川、永坂監修：よくわかる数値計算・アルゴリズムと誤差解析の実際、p. 72、2004、日刊工業新聞社。
- [5] 瀬島、田中、坪井：PC クラスタを用いた電磁界解析、電気学会静止器・回転機合同研究会資料、SA-05-11・RM-11-79、2005.
- [6] 瀬島、田中、坪井：並列処理を用いた電磁界解析、電気学会静止器・回転機合同研究会資料、SA-05-72・RM-05-79、2005.
- [7] H. Tsuboi, M. Tanaka, K. Ikeda, K. Nishimura: Computation results of the TEAM Workshop Problem 7 by finite element methods using tetrahedral and hexahedral elements, *Journal of Materials Processing Technology*, Vol. 108, pp. 237-240, 2001.