オブジェクト指向設計による並列フリーメッシュ法ソフト開発*

仲 間 豊^{*1}, 関 東 康 祐^{*2}
安 藤 知 明^{*3}, 矢 川 元 基^{*4}

Object Oriented Development of Parallel 3D Free Mesh Method

Yutaka NAKAMA^{*5}, Yasuhiro KANTO,
Tomoaki ANDO and Genki YAGAWA^{*1} Fuji Research Institute Corporation,
2-3 Kanda-nishiki-cho, Chiyoda-ku, Tokyo, 101-8443 Japan

Free Mesh Method (FMM) is a kind of node-based finite element method, which does not require mesh information. Therefore it has a main advantage of meshless method; cutted cost of calculation by omitting mesh creation. Then it seems suitable for an adaptive method, a crack propagation problem, and a large-scale problem. Moreover, it still has merits of finite element method, i.e., it can use many existent techniques developed for finite element method. Although FMM has been demonstrated to be applied to many fields of research, no general purpose program based on FMM has been developed yet. So we designed Free Mesh Method program for 3D problem. Some of main features of this software are: (1) Large-scale: It can be executed in parallel environment, such as PC cluster with domain decomposition method (DDM). (2) Expandable: It is designed by object-oriented approach and new functions can be added very easily. Here we describe the local mesh generation of FMM, generalized recursive bisection method for DDM, the object-oriented design of the software, and the object-oriented parallel matrix solver.

Key Words: Computational Mechanics, Finite Element Method, Free Mesh Method, Domain Decomposition, Parallel Computing

1. ま え が き

最近の解析技術の技術革新に伴い、複雑な3次元形状や大規模問題を効率良く解析をする事が求められている。従来のFEMでは、(1)複雑な3次元形状の問題に関してはメッシュ生成に要する人的作業量が膨大になること、(2)大規模問題に対しては必要メモリや計算時間の面で大量のコンピュータ資源を必要とすることが課題となり、実用的に十分と言えない状況にあった。また、(3)プログラムに新規の解析機能を追加するなどの機能アップには、チェックを含めて多大な工数を要するため、システム発展の阻害となっていた。

これらの問題の解決法として、ユーザーがメッシュ生成を必要としない手法の一つとしてのフリーメッシュ法 (Free Mesh Method, FMM)⁽¹⁾⁽²⁾の研究と、大規模問題に対しては、メモリと計算時間を縮減する領域分割法に基づく並列計算の研究が進んでいる。⁽³⁾⁽⁴⁾ わ

れわれは、この二つの研究成果に加えて、オブジェクト指向技術を採り入れることにより(3)のシステム発展阻害要因を取り除いた、領域分割法を用いたフリーメッシュ法並列基盤ソフトウェア:FMM3Dの開発を行った。本ソフトウェアは、98年度IPA (Information-technology Promotion Agency, Japan)の「高度情報化支援ソフトウェア育成事業」の支援を得て開発したプログラムを基盤としている。

本ソフトウェアは、節点及びメッシュ生成が不要であること、分散および共有メモリ型並列計算機、あるいはEWSを結合したクラスター環境において、大規模な問題に対しても適用可能であることが特徴であり、さらに、オブジェクト指向技術を採り入れることにより、利用者が容易にさまざまな解析機能を組み入れる機構を備えている。本ソフトウェアを利用すれば、開発者がプログラムの既存部分を全く変更せずに、簡単に独自の解析機能(例えば流体解析等)を組み込むことができ、しかも、並列化のことを何一つ意識せずに自動的に並列計算用プログラムを作成することができる。

* 原稿受付 2003年4月30日。

^{*1} 正員、(株)富士総合研究所(☎101-8443 東京都千代田区神田錦町2-3)。^{*2} 正員、豊橋技術科学大学機械システム工学系(☎441-8580 豊橋市天伯町雲雀ヶ丘1-1)。^{*3} (株)先端力学シミュレーション研究所(☎351-0198 和光市広沢2-1 理化学研究所研究交流棟W109)。^{*4} 正員、東京大学大学院工学系研究科(☎113-8656 東京都文京区本郷7-3-1)。

E-mail: nakama@garlic.q.t.u-tokyo.ac.jp

2. FMM の概要

フリーメッシュ法 (FMM) は、有限要素法 (Finite Element Method, FEM) をベースとしているが、メッシュデータが不要な手法である。また節点データも内部で自動生成する事により不要とすることができる。FMM は、メッシュデータ不要という意味ではメッシュレス法と言えるが、自動メッシュ生成機能付きの FEM の一種であり、その特徴は各節点毎にローカルなメッシュを作成していくというメッシュ生成法にある。

FMM の基本的な考え方を示す例として 2 次元の場合を示す。まず各節点毎に、その節点を中心節点 (central node) として、他の節点 (衛星節点, satellite node) と中心節点から、一時的に仮要素 (ローカルメッシュ, local mesh) を作成し、これらの要素剛性行列から中心節点に作用する成分を求め、全体行列を作成する。具体的には、Fig.1 のように各節点毎に、その節点を中心節点 l として、衛星節点 (m, n, o, p, q, r) を集め、一時的にローカルメッシュを作成する。ローカルメッシュの各要素 (lmn など) に対して、要素剛性行列 $[K]_{lmn}$ を作成し、これらの要素剛性行列から中心節点 l に作用する成分を求め、行の各成分を足し込む。これを全節点について実施すると、解くべき剛性行列全体が得られ、それを行列解法により解く。

Fig. 1 Local Mesh

3. ローカルメッシュ生成

FMM では、仮想的なローカルメッシュを生成する必要があるが、その手法としては、逐次添加法による Delaunay 分割法や包装法が用いられている。

しかし、ローカルメッシュ生成段階で、例えば Delaunay 分割法をそのまま適用すると、グローバルなメッシュの整合性が確保できないことや、縮退した要素を作成するという問題が起こる可能性がある。⁽⁵⁾

これらは、局所要素生成の非一意性問題と、数値誤差に起因している。

3.1 整合性の確保 簡単のため 2 次元の場合を例にとりて Delaunay 分割の非一意性を説明する。Delaunay 分割は各要素の外接円内に他の節点が存在しないように分割を行う。しかし、同一円周上に 4 つ以上の節点が存在する (縮退している) 場合には、その節点によって構成される要素は、節点座標からは一意には決定できない。例えば Fig.2 の Case A と Case B はどちらも Delaunay 分割の条件を満たしている。FMM でローカルメッシュを作成する場合、例えば、節点 A が中心節点の場合には Case A のように要素が生成され、一方、節点 C が中心節点の場合には Case B のように要素が生成されるという事が起こりえる。FEM 用のメッシュ生成のように、解析領域全体の節点を用いて Delaunay 分割をおこなう場合には、このような非一意性の問題は生じない。また、数値誤差によって、4 節点が同一円周上に存在するかどうかなどの判定が、中心節点がどの節点になっているかの違いによって異なる結果を生じため、整合性を失ったり、縮退した要素を作成することがある。

Fig. 2 局所要素生成の非一意性

我々は、Delaunay 分割法におけるローカルメッシュの生成過程において、次のような制限を設けてメッシュ生成処理を行うことにより、局所要素の非一意性問題、および数値誤差問題の回避に成功した。

- ・ (グローバルな番号付け): 節点には全体として一意的な番号を付け、各中心節点のローカルメッシュ生成の際には、常にその番号順に節点を配置し、Delaunay 分割を形成する。
- ・ (一意性の確保): Delaunay 分割中に、縮退関係が起こった場合の取り扱いを一意にする。つまり新たに設置した節点がある要素の外接円上に存在する場合、その要素を削除して作り直すか、そのまま残すかのどちらかに統一する。
- ・ (数値誤差による誤判定の排除): 4 節点が同一円上に載っているかどうかなどの判定を行う際には、

常に演算順序が常に等しくなるようにする。つまり、足し算やかけ算の順番をグローバル番号が小さい順あるいは大きい順におこなうようにする。

3.2 メッシュ生成の並列化 FMMにおけるメッシュ生成法は並列化に向いている。FMMにおけるローカルメッシュ生成の際に必要な情報は、近傍節点の座標情報のみである。このためこのメッシュ生成処理を並列でおこなった場合でも、通信を全く必要としない。ローカルに作成したメッシュがグローバルに整合性を保つため、FMMにおけるメッシュ生成の並列化効率は、常にほぼ100%の効率を確保できる。

4. 領域分割法

並列処理の高効率化のためには、各PEの処理量を均等にする事と、通信量を減らす事が重要である。FMMでは、メッシュ情報を持っていないため、領域分割法の手段に制約をうける。ここでは、領域分割法のひとつであるリカーシブバイセクション法を採用しており、各領域に含まれる節点の数を等しくする事により各PEの処理量を均等化し、分割面の面積を少なくする事によりPE間の通信量を減らしている。リカーシブバイセクション法は、解析領域の長軸方向に対して、それぞれの領域に含まれる節点が等しくなるように領域を2分割し、この処理を望ましい分割数が得られるまで再帰的に繰り返す手法である。Fig. 3は、PE数が4つの場合のこの手法による分割例である。

Fig. 3 Decomposition of 2D

領域形状が横長であるため、最初に(1)の分割が行われ、次に分割された領域が縦長になるため(2)による分割が行われる。領域分割数とPEの数が一致するためここで分割が終了する。通常、この方法を採用した場合、PE数が2ⁿに制約されてしまうが、以下のアルゴリズムを再帰的に行なう事により任意数の領域への分割が可能となる。

「各領域が自分自身を m 個の領域に分ける必要がある時、 $m_1:m_2$ ($m_1 = \lfloor m/2 \rfloor, m_1 + m_2 = m$) の節点数の

比になるように二つの領域に分割を行って行く」

これによって、任意のPE数で並列計算が行える。Fig. 4に3次元の立方体を7分割した例を示す。7分割の場合には上記アルゴリズムを3回繰り返すことになるが、図中の境界面の辺上にある番号(1),(2),(3)は、その分割順を示している。

Fig. 4 Decomposition of 3D-cube

5. オブジェクト指向設計

本ソフトウェアでは、オブジェクト指向による設計をおこなうことで、将来的にプログラムの機能拡張・変更を容易とする仕組みを取り入れた。また、単CPU用のプログラムを拡張するだけで、並列化プログラムの自動作成する機構を開発した。

5.1 FMM3Dにおける機能追加法 本ソフトウェアの初期バージョンでは、静的応力解析と熱伝導解析のみが可能だが、これに動的解析や、流体解析などの機能を拡張する場合には、Analysisクラス、要素方程式クラス、材料クラス、境界条件クラスの4つのクラスのいくつかのメソッドを継承するコードを追加するだけで良く、既存のソースコードを一切修正する必要が無い。

Figure 5でグレーで示しているクラスの横および下の「●●」で示した(4ヶ所の)部分に追加してクラスを作成する。また、Table.1に、機能追加の際に関連するメソッド一覧を示す。“必須?”のカラムにある△と○は、メソッドの継承の必要性を示しており、○は必ず継承する必要があるメソッド、△は解析種別によっては継承が不要となるメソッドである。わずかなコードの追加で機能拡張が可能となることがわかる。

Fig. 5 Class diagram

例えば、熱伝導解析を追加した際には、既存の4つのクラスに新たに4つのクラスを継承機能を利用して追加し、コメント行を除き約600行要した。これは、継承を利用しない場合に比べ、約1/3の行数である。

Table 1 Additional source code of methods

Class	Method	必須?	メソッドの機能概要
Analysis	pre	△	計算ループに入る前の処理
Analysis	loop	△	各ループでの処理(剛性行列作成+行列計算)
Analysis	lpost	△	各ループ後の処理
Analysis	post	△	計算終了後の処理
Analysis	stans	△	解ベクトルを各節点に設定
Analysis	mkmst	△	剛性行列を作成する処理
Analysis	solve	△	剛性行列を解く処理
Analysis	output	△	計算結果をファイルに出力する
Material	readfp	○	ファイルからの材料データの読み込み
Boundary	readfp	○	ファイルからの境界条件データの読み込み
Ematrix	mkmst	○	要素剛性行列Kの作成
Ematrix	mktvec	○	要素節点ベクトルrの作成
Ematrix	bcdirc	○	要素上にある接点のディレクショナル境界条件を設定
Ematrix	bceum	○	要素上にある接点のノーマル境界条件を設定

5.2 並列化プログラムの自動作成 メインの開発言語にC++を用いており、並列化にはMPI(Message Passing Interface)を用いている。並列化の部分は、これらの解析機能関連のクラスとは完全に独立であり、機能拡張の際には並列化のことは一切気にする必要が無いように設計されている。そのため、1 CPU用の解析プログラムのみに留意して解析機能の追加を行っても、1 CPU用のモジュールを作成(make)した時に、自動的に並列用のモジュールも作成される。完全自動の並列化プログラム作成機構となっている。

6. オブジェクト指向行列ソルバー

6.1 行列解法とその並列化手法 行列ソルバーは、対称行列用の反復解法である前処理付き共役勾配法(CG法と記す)採用している。本来、反復法であるCG法そのものの並列化は容易に可能であるが、収束を加速するためにおこなう前処理には並列化が容易

なものと困難なものがある。ここでは、並列化が容易となるように、前処理法の逆行列を多項式近似することにより逐次処理を不要とする方法を採用した。

6.2 行列ソルバークラス 今回開発したCG法による行列ソルバーは、それ自体をメソッドとして含む行列ソルバークラスという形で実現されており、単CPU版の行列ソルバークラスを並列版のソルバークラスが継承する形で実現されている。この行列ソルバークラスの一番の特徴は、これを使用するプログラム開発者がクラス内部のソースを一切読まなくとも利用可能な点にある。また、インターフェースとなる各メソッドの引数には配列を一切含まないため、行列のデータ構造を知る必要が無い。並列計算のための通信データの事も一切考える事も無しに、行列演算が行える。実際、 $Ax = f$ の方程式を解く場合には、Table 2にあるメソッド等を利用するだけで、行列演算が行える。

2カラム目は並列版の行列ソルバークラスメソッド一覧である。ほとんどのメソッドは単CPU版を継承するだけで、単CPU版との違いは、(1)minitでのメモリ確保時に全PEでの総節点数Naを教える必要があること、(2)setneメソッドで各節点のグローバルな節点番号を覚えておく必要があること、(3)setpeメソッドで各節点の担当PE番号を覚えておく必要があることの3つである。通信に必要な袖領域の情報は、クラス内部で自動作成する。

また、これらのメソッドの大部分はC++で開発されているが、行列計算を行う部分(msolve)はFortranで書かれているため、計算速度低下の問題は無い。

6.3 並列化効率 FMM3Dのソルバープログラムで静的応力解析を行った際の並列化効率を示す。Fig. 6は、Linux PCのクラスタ環境において、9500接点のモデルをそれぞれ1,2,4,6 CPUで計算計算した時の並列化効率を示すグラフである。小さなモデルでも並列化効率はきわめて高いことがわかる(2CUPで96.9%, 4CPUで94.4%, 6CPUで92%)。これは、行列計算の部分がFortranで書かれており、その部分の並列化効率が高いことにもよるが、FMMの利点である「メッシュ生成に通信が必要ない」ことも大きな理由である。

7. 解析例

FMMソフトウェアを用いた解析例を示す。FMMではメッシュデータおよび節点データは入力データとしては必要としないため、CAD等で作成した形状データに境界条件を与えるだけで解析を行なう事が可能となる。以下に解析の流れを示す。

1. 形状データを作成する。形状データはCAD等を

Table 2 Methods in matrix solver class

Single ver.	Parallel ver.	Function
minit(N,F)	minit(N,F,Na)	Initialize (N:num.of nodes, F:degree of freedom)
mclear()	mclear()	zero clear of A and f.
maxc(N)	←	Set Max Iteration count.
resd(R)	←	Set Residual to R.
setk(i,j,R)	←	Set i,j-element of A to R.
setf(i,R)	←	Set i-th element of f to R.
addk(i,j,R)	←	Add R to i,j-element of A.
addf(i,R)	←	Add R to i-th element of f.
dirc(i,R)	←	set Dirichlet condition (modify A and f)
msolve()	msolve()	solve and calculate x.
geta(i)	←	get i-th element of x
—	setne(n,i)	Set n-th node's global node number to i.
—	setpe(n,i)	Set n-th node's PE number to i.

Fig. 6 Parallel Efficiency

利用して作成したものを利用する。Fig.7の(1a), (1b)は解析領域の形状を可視化したものである。(1b)はteddy⁽⁶⁾を用いて作成した。

- 境界条件を設定する。境界条件はユーザーが設定する必要がある。Fig.7の(2a), (2b)は境界条件の設定状況である。
- 節点を自動生成する。Fig.7の(3a), (3b)は自動生成を行なった例である。
- 解析を行なう。Fig.7の(4a)は静的応力解析を行なった結果の可視化例, (4b)は熱伝導解析を行なった結果の可視化例である。

本ソフトウェアでは、IGES形式やVRMLなどCADで作成されたデータの他に、手書きによる3次元形状生成ツールteddyのデータをそのまま形状データとして取り込むため、3次元のCADデータを作成せずとも手軽に数値計算が行える。

8. む す び

3次元FMMソフトウェアの設計/開発を行い、

- ・FMMは並列化効率が高いことを示した(4CPUで94.4%)。これは、メッシュ生成の並列化効率が

100%であることなどによる。

- ・オブジェクト指向を取り入れることにより、機能拡張の容易性を実証した。

また、解析機能が簡単に組み込めるだけでなく、並列計算用のプログラムを自動で生成することができる。これらの特徴のため大規模・複雑な問題、及び各種問題への展開が容易となる。数値解析分野でオブジェクト指向技術を本格的に取り入れた例は少ないが、数値解析モデルは多種多様であるため、この技術は有効と考えられる。

謝 辞

本ソフトウェアは、1998年度IPAの「高度情報化支援ソフトウェア育成事業」における「領域分割フリーメッシュ法による高度設計基盤ソフトウェアの開発」において開発したプログラムを基盤とし、「フリーメッシュ法研究会」で継続研究した成果をもとに改良を加えたものである。「領域分割フリーメッシュ法による高度設計基盤ソフトウェアの開発」および「フリーメッシュ法研究会」に参加して頂いた研究者から貴重な助言を頂いたことを感謝致します。

文 献

- (1) 矢川元基, 山田知典, 河合浩志, メッシュレスFEMについて, 機械学会第72期講演会論文集, (1995), p.210.
- (2) G.Yagawa, T.Yamada, *Free Mesh Method: A new meshless finite element method*, Computational Mechanics, 18, (1996), p.383.
- (3) Genki Yagawa, Tomonori Yamada, *Parallel Computing with Free Mesh Method*, Transactions of the SMIRT 14, Lyon, France, August 17-22, (1997)
- (4) 矢川元基, 塩谷隆二, 超並列有限要素法, 朝倉書店, (1998)
- (5) 細川孝之: フリーメッシュ法の3次元化に関する研究, 東京大学大学院工学系研究科情報工学専攻修士論文, (1998)
- (6) Takeo Igarashi, Satoshi Matsuoka, Hidehiko Tanaka, *Teddy: A Sketching Interface for 3D Freeform Design*, ACM SIGGRAPH'99, Los Angeles, 1999, pp.409-416.

(1a) 形状データ 1

(1b) 形状データ 2

(2a) 静的応力解析用境界条件

(2b) 熱伝導解析境界条件

(3a) 節点生成例 1

(3b) 節点生成例 2

(4a) 静的応力解析結果

(4b) 熱伝導解析結果

Fig. 7 flow of analysis