オブジェクト指向型有限要素モデラーによる 船体メッシュ生成の並列化

正員 川 村 恭 己* 正員 大 坪 英 臣**
正員 鈴 木 克 幸**

Parallel Mesh Generation of Ship Structure
Using Object Oriented Finite Element Modeling System

by Yasumi Kawamura, *Member* Hideomi Ohtsubo, *Member*
Katsuyuki Suzuki, *Member*

Summary

In dealing with structure of complex configuration such as ship structure, generation of finite element (FE) model is a major bottleneck. To overcome this problem, object-oriented FE modeler, MODIFY, has been developed by the authors^{(1),(2),(3)}. When using MODIFY, user defines and assembles geometry objects, analytical condition objects and relation objects. Then MODIFY generates mesh for each geometry objects independently. For this reason, MODIFY can easily generate mesh for complicated configuration like ship structures.

In this paper, parallelization using engineering work station (WS) on network environment is studied in order to enhance the speed of mesh generation. This concept is incorporated in MODIFY based on client-server model that is a basic idea for network application. The efficiency of parallel mesh generation is shown with several examples.

1. 結 言

近年、船体構造設計の分野は大きな発展を遂げ、有限要素法による大規模構造解析が数多く行なわれるようになってきた。しかしながら、有限要素法解析においては解析そのものよりも解析モデル生成が解析作業全体の大きなボトルネックとなっている。著者らは、この解析モデル生成の自動化・省力化を目的として、現在までにワークステーション (WS) 上で動作するオブジェクト指向型有限要素モデラー MODIFY の研究・開発を行ってきた^{(1),(2),(3)}。MODIFY では、解析対象全体を、力学/工学上意味のある部品オブジェクトがそれらの幾何学的関係を表すリレーションで結合しているという Entity-Relationship モデルで表現することにより解析者のメンタルモデルと実際の解析モデル表

現間の溝を減らし、解析対象の問題定義及び修正を容易にしている。また、これによって有限要素法モデル生成そのものを容易にしているという特徴がある。

また一方、最近では WS によるネットワーク環境が発展し、ネットワークを利用した並列処理の行なえる数多くのソフトウェアが使われるようになってきている。有限要素モデラーにおいてもネットワーク環境を利用することによって、システム全体の効率やシステムの使いやすさ (マンマシンインタフェース) を向上することが可能であると考えられる。例えば、単一のユーザインタフェースによりモデリングから解析まですべてを行う場合、大規模解析には大型計算機などのより性能の良い計算機を用いた方が望ましいのでネットワークの利用は不可欠となる。さらに、MODIFY では各形状オブジェクトごとに独立にメッシュ生成を行なうことができるために、複数の WS を用いてメッシュ生成を行なえば、メッシュ生成時間の飛躍的な高速化を行うことができると思われる。特に、船体構造のような数多くの部品から成るモデルにおいてはこの効果は大変大きいと考えられる。

本研究では、曲面板組構造を対象としたオブジェクト指

* 東京大学大学院工学系研究科

** 東京大学工学部船舶海洋工学科

向型有限要素モデラー MODIFY において、クライアントサーバモデルの考え方をを用いて複数の WS を用いた並列処理メッシュジェネレーションを実現した。

2. 船体構造解析におけるメッシュ生成

近年、船体の構造設計において直接強度計算を行なう場合には、全船解析を始めとする有限要素法による大規模な応力解析がしばしば行なわれるようになってきている⁴⁾。船体構造は板が複雑に結合した構造を持っている。そのため、有限要素メッシュを生成する際には板の結合部において節点を発生した正しい節点・要素の結合状態を定義する必要がある。Fig. 1 に、船体構造の一例としてバルクキャリアの内部構造を示す。この例では、例えば内底板は多くのガーダやフロアによって複数の境界で区切られるために、単に内底板でメッシュの生成を行なうだけでなく、他の板との結合状態を考えながらメッシュ生成を行なう必要がある。

曲面板組構造を対象とするオブジェクト指向型有限要素モデラー MODIFY²⁾ においては、これら一つ一つの区切られた領域をシェルオブジェクトとして定義し、それぞれのシェルオブジェクトで独立にメッシュを生成した。まず、CAD より IGES 形式のファイル⁵⁾ を出力する。そして、MODIFY の IGES ファイルの自動読み込み機能²⁾ によってこれを読み込み MODIFY 内に形状が読み込まれシェルオブジェクトが定義される。複雑なモデルを対象とする場合には、大変多くのシェルオブジェクトが定義されることになる。次に、ユーザは MODIFY 本体のユーザインタフェースを通じて、解析条件等を定義しながら解析対象の問題

Fig. 1 Example of Ship Structure (Bulk carrier)

定義を行なう。そして、ユーザが要素の大きさの基準値を与えると MODIFY が自動的にメッシュを生成する。

MODIFY におけるメッシュ生成のアルゴリズムは 2 次元のフロント法を曲面用に改良したアルゴリズムを用いている (Fig. 2)。その概要は、(1) 各シェルオブジェクトの結合部で整合のとれたメッシュの生成を行なうためにシェルオブジェクトの境界辺上に共通の境界節点を生成、(2) それをもとにシェルオブジェクトの面上に内部節点を生成、(3) 得られた節点を結ぶ、という手順で行なわれる。節点を結ぶアルゴリズムは曲面用に改良したフロント法を用いている²⁾。そして、(2)、(3) の処理をそれぞれのシェルオブジェクトで逐次的に独立に行っている。

従来、船体構造を有限要素法により構造解析する場合解析自体が数日で終了するのに比べて有限要素モデル生成に場合によっては数カ月程度かかっていた。本研究では、これらのメッシュ生成を並列処理的に行ない、メッシュ生成の高速化を行なうことを提案する。これによって、MODIFY では船体構造のような複雑な有限要素モデルの生成が、CAD によるモデル生成が一日程度、メッシュ生成が数時間程度で行なえることが期待できる。また、その実現法としては WS におけるネットワーク環境を利用する。

並列的にメッシュを生成を行なうことを考えた場合、(1) の過程はシェルオブジェクト間で要素間の整合をとるため

Fig. 2 Mesh Generation Procedure in MODIFY

に、一度に行なう必要がある。また、メッシュ生成全体にかかる時間の中で(1)の過程にかかる時間は(2),(3)にかかる時間に比べて無視できる。そこで、並列的に行う処理として(2),(3)を複数のWSで行なうことになる。以下の説明では、(2)(3)の過程をメッシュ生成と呼ぶ。

次節では、まずネットワークの利用法について述べる。

3. 並列処理メッシュ生成におけるネットワーク利用法

コンピュータシステムでネットワーク機能を利用するために、現在では大型からWSまでのあらゆる種類のコンピュータ上でTCP/IP(Internet プロトコル)と呼ばれる通信プロトコルが実装されるようになってきている^{6),7)}。本研究では、WS上で並列処理メッシュ生成を行なうために、TCP/IPプロトコルを用いて異なるWSのプロセス(Unixにおけるプログラム実行の基本単位)間で通信を行なう。その際、パケットソケットと呼ばれるC言語によるインタフェースを用いる。また、複数のWSを用いるためにメッシュ生成をコントロールする必要があるが、その際クライアントサーバモデルを利用した。

3.1 クライアントサーバ (client-server) モデル

複数のWSを利用してメッシュ生成を行う場合、MODIFYから他のWS上のメッシュジェネレーションを起動しなくてはならない。また、あるWSでメッシュ生成が終了したら、そのWSへ次のメッシュ生成を割り当てるなど、メッシュ生成全体をMODIFYでコントロールする必要がある。すなわち、プロセスのコントロールを行う必要があるわけだが、この場合クライアントサーバ (client-server) モデルの考え方が有効である。

クライアントサーバモデルはネットワークアプリケーションにおける標準的なモデルである。サーバとはクライアントプロセスからの接続を待ちクライアントのために何かをするためのプロセスである。また、クライアントとはサーバに何らかの処理を要求するためのプロセスである。一般的なクライアントサーバモデルは以下のような手順になる。

1. あるコンピュータ上でサーバプロセスが起動される。サーバは初期化された後、クライアントプロセスがサービスを求めて接続して来るまで待ち状態になる。
2. 同じコンピュータ、またはネットワークで結合された他のコンピュータ上でクライアントプロセスが(多くの場合ユーザによって)起動される。クライアントプロセスはある種のサービスに対する要求をサーバプロセスに送る。
3. サーバプロセスはサービスを処理した後、結果をクライアントプロセスに返す。サーバプロセスは再び待ち状態に入り、次のクライアントが接続して来るまで待つ。

Fig. 3 Inter Process Communication Using Stream Socket

Fig. 3 は、この典型的なクライアントサーバモデルによるプロセス間通信の様子を示した図で、プログラム内で用いるシステムコールを同時に示してある。まず、サーバでは使用したい通信プロトコルのタイプを指定する。そして自分のアドレスを指定しクライアントから要求を受けるための準備を行なう。接続が可能であることをコンピュータに示した後に、クライアントからの接続を待ち状態に入る。一方クライアント側では、通信プロトコルのタイプを指定した後、接続するサーバを指定しながらサーバとの接続を確立する。接続が確立した後は、要求データがクライアントからサーバに送られる。そして、サーバで何らかの処理が行なわれた後にクライアントへと応答データを返すこととなる。

また、サーバプロセスはさらに次の2種類に分けることができる。

反復サーバ (iterative server) サービスが短時間でできる場合や、長時間かかっても複数のプロセスを起動する必要がない場合にはサーバプロセス自身が要求に対する処理を行う。

並行サーバ (concurrent server) 要求の処理時間が不定である場合などで要求を並列的に処理したい場合は並行サーバを用いる。並行サーバはクライアントからの要求に対し、他のプロセスを呼び出しその処理を任せる。これによって、並行サーバは再び待ち状態に入ることができ、他のクライアントからの要求を受け付けることができるようになる。

4. 並列処理型メッシュジェネレータの設計

本研究では、クライアントサーバモデルを用いて MODIFY 内に並列処理型メッシュジェネレータを実現した。以下にその詳細について記す。

4.1 システム全体像

まず、各 WS でメッシュを生成するために各シェルオブジェクトのメッシュ生成領域に関するデータ（境界節点のデータ・曲面のデータ・メッシュの疎密をコントロールするための力学的重要度のデータ）を MODIFY 本体からそれぞれの WS へと転送する必要がある。また、メッシュ生成の要求を受けた WS では、送られたデータに関してメッシュを生成し、メッシュ生成結果（節点・要素のデータ）を MODIFY 本体に転送する必要がある。そこで、本研究では Fig. 4 のようにメッシュ生成の際には、生成を行なう WS 上に実際にメッシュを生成するメッシュ生成プログラムサーバプログラム（メッシュジェネレータサーバ（以下 MG サーバと呼ぶ））を配置（起動）する。そして、MODIFY 本体はメッシュ生成を各 WS 上の MG サーバに要求するクライアントとなる。すなわち、前節で説明したクライアントサーバモデルになる。

4.2 MG サーバ

一つの WS に過度な負担をかけてメッシュ生成の処理時間が遅くならないように、各 WS には一つのメッシュジェネレーションが起動するのが望ましい。そこで、メッシュジェネレータは一度に一つの要求のみを処理する反復サーバとして実現する。具体的には、次のようなステップを持つループを繰り返すことで、MODIFY 本体の要求にしたがってメッシュ生成を行なう（Fig. 5）。

1. ループに入る前にネットワーク使用の準備を行なう（socket, bind, listen システムコール）。
2. MODIFY 本体からの要求を待つ（accept）。
3. MODIFY 本体から初期データを受信する（read）。
4. メッシュ生成処理を行なう。
5. 生成結果を MODIFY 本体へ知らせるため結果の送信

Fig. 4 The Conceptual Image of Parallel Mesh Generator

Fig. 5 Flow Chart of Mesh Generation Server

を行う（write）。

メッシュ生成処理（ステップ4）は、MODIFY 本体から送信されてきた境界節点のデータ・曲面の形状データ・重要度に関するデータをもとに、シェルオブジェクトの面上に内部節点を生成し、得られた節点を結ぶという手順となる²⁾。

4.3 MODIFY 本体の役割

MODIFY 本体は複数の WS 上にある MG サーバに対してクライアントとなる。MODIFY 本体はメッシュ生成全体をコントロールする必要があるが、そのアルゴリズムは Fig. 6 のようである。まず最初に、使用する WS のデータ及び各シェルオブジェクト毎にメッシュ生成領域に関するデータを生成しておく。その後に、Fig. 6 のようなループへと入っていく。ループ内では、

1. メッシュ生成を行なうデータがあれば他の WS 上の MG サーバへデータの転送、
 2. メッシュ生成結果が来ていれば他の WS 上の MG サーバから計算結果の受け取り、
 3. すべてのメッシュ生成が終了していれば完了、
- という3つのステップを1, 2, 3の順の優先順で行なっている。

この時、具体的には、ステップ1ではまず MG サーバとの通信回路の結合を行なった後に（socket, connect システムコール）、データを転送する（write）。ステップ2では MG サーバからメッシュ生成結果の送信がないか、どの WS の MG サーバからメッシュ生成結果の送信があったかどうかのチェックを行ない、送信があればメッシュ生成結果を受

Fig. 6 Flow Chart of MODIFY for Mesh Generation Control

信する(read)。なお、メッシュ生成結果の送信の有無等の判断はselectシステムコールを用いている。

さらに、MGサーバがマシントラブル等によって動作不能になった場合もステップ2でチェックが行なわれメッシュを再生成するための操作が行なわれる。

5. 実行結果・考察

上記のようなモデルに従って並列処理型メッシュジェネレータをMODIFY内に実現した。以下の実行結果では、MODIFY本体はWS (SPARC station 2) 上で動作させ、MGサーバは複数のWS (SPARC station 10) 上に配置した。

5.1 船首部を簡略化したモデル

まず、Fig. 7のような船首部を簡略化した構造(Fig. 7は外板だけを表示)のメッシュ生成を行なった結果を示し、一台のWSを用いてメッシュを生成した場合との比較を行なう。Fig. 8にMODIFYにおけるシェルオブジェクトの定義を示す。この例ではメッシュ生成の単位となるシェルオブジェクトが47個存在している。

前節で示したように、メッシュ生成前には使用するWSにMGサーバを走らせる。そして設定ファイルに使用するWSの一覧を記述しておく。その後、MODIFYよりメッシュ生成を行なうと設定ファイルに従って、それぞれのシェルオブジェクトのメッシュ生成を自動的にWSへと振り分けていく。振り分けのアルゴリズムは、すべてのWSが同等の性能を持つと考え、メッシュ生成時間の多くかかろうなシェルオブジェクト(境界節点の数で判断している)から設定ファイルの上位のWSへ次々と割り当てていく方法を用いている。また、メッシュ生成を行なうマシンの性能が異なる場合にもより速いマシンを設定ファイルの上位に記すことによって、効率的にメッシュ生成を振り分け

Fig. 7 Simplified Bow Structure (CAD Model)

ることができる。

メッシュ生成結果はFig. 9のようになる。このメッシュの節点数は2216個、要素数は2581個である。WSはws 1, 2, ..., 10と名付けられたSPARC station 10を、合計10台利用した。Fig. 10に各WSにおけるメッシュ生成の様子を示す。図では、横軸が時間の経過(秒)を示し、縦軸はそれぞれのWSを表している。全体のメッシュの生成にかかった時間は約350秒(5.83分)である。それぞれのWSにおいて一本の矢印が一つのシェルオブジェクトのメッシュ生成にかかった時間を示している。このメッシュ生成の場合、従来の複数のWSを使用しないメッシュ生成の場合WS (SPARC Station 10) で4282秒(約70分)かかっている。この結果を見ると、並列的にWSを利用した分だけメッシュ生成に高速化が行なわれたことが分かる。

5.2 バルクキャリアの中央部

次に、Fig. 11のようなバルクキャリアの中央部の簡易モデルをメッシュ生成した例を示す。Fig. 11は、外部CAD (Pro/Engineer) により生成されたモデルである。Fig. 12にMODIFYにおけるシェルオブジェクトの定義を示す。メッシュの生成単位であるシェルオブジェクトは78個定義されている。

メッシュを生成した結果をFig. 13に示す。このメッシュ生成も船首部の例と同様に10台のWS (SPARC Station 10) を用いて生成された。節点数は14658個、要素数は14930個である。それぞれのシェルオブジェクトのメッシュ生成はFig. 14に示したように各WSに振り分けられている。メッシュ生成にかかった時間は3200秒(約53分)

Fig.8 Simplified Bow Structure (Defined by Shell Objects in MODIFY)

Fig.9 FE Model for Simplified Bow Structure

Fig. 10 Time History of Mesh Generation for Each WS (Simplified Bow Structure)

である。Fig. 14を見ると、WS 1に割り当てられたシェルオブジェクトのメッシュ生成が全体にかかる時間を支配していることが分かる。これは、Fig. 11のバルクヘッド部分のモデル化を細かい部分まで行っていないので、その部分のシェルオブジェクトが他のシェルオブジェクトに比べて大きな面積を持ち、より多くのメッシュ生成時間がかかったためである。この結果から考えると、各メッシュ生成領域の大きさに大きなばらつきがある時は、メッシュ生成時間の短縮効果はそれほど期待できないと考えられる。し

Fig. 11 Bulk Carrier (CAD Model)

かしながら、船体構造のような多くの部品から成る複雑なモデルでは、ある一定の大きさのメッシュ生成領域が多くあるために並列処理のメッシュ生成は適していると言える。

Fig. 12 Bulk Carrier (Defined by Shell-Objects in MODIFY)

Fig. 13 FE Model for Bulk Carrier

Fig. 14 Time History of Mesh Generation for Each WS (Bulk Carrier)

6. 結 言

本研究では、クライアントサーバの考え方に基づく、複数のWSを利用した並列処理型メッシュジェネレータの設計を行ないこれをオブジェクト指向型有限要素モデラーMODIFY上に実現した。これによってメッシュ生成時間を大幅に短縮することができた。また、この考え方が船体のような多くの部品からなる構造物には特に有効であることが示された。

参 考 文 献

- 1) 大坪英臣, 久保田晃弘, 川村恭己, オブジェクト指向に基づく船体構造解析用 FEM モデリングシステム, 日本造船学会論文集, 第 170 号, 1991
- 2) 大坪英臣, 久保田晃弘, 川村恭己, 平木常正, 斉藤正樹: オブジェクト指向型有限要素モデラーの曲面板組構造への適用, 日本造船学会論文集, 第 172 号, 1992
- 3) 大坪英臣, 川村恭己, 久保田晃弘: オブジェクト指向データベースによる有限要素解析モデルの管理, 日本機械学会第 5 回計算力学講演会講演論文集, pp. 17-18, 1992
- 4) 倉本美男, 戸沢 秀, 白木原浩, 井上俊司, 伏見 彬: 波浪中の船体局部応力のシミュレーション手法に関する研究, 日本造船学会論文集, 第 170 号, 1991
- 5) Initial Graphics Exchange Specification (IGES), Version 3.0, U.S. Department of Commerce, April 1986
- 6) W. Richard Stevens 著/篠田陽一訳, UNIX NETWORK PROGRAMMING, Prentice Hall, トッパン, 1990
- 7) Douglas Comer 著/村井純, 楠本博之訳, TCP/IP によるネットワーク構築—原理・プロトコル・アーキテクチャー, Vol. 1, bit 別冊, 共立出版, 1991