並列境界要素解析フレームワークの設計と実装

野瀬田 裕樹[†,†††] 河合 直聡[†,†††] 岩下 武史[†,†††]
高橋 康人[††,†††] 美舩 健[†,†††] 中島 浩[†]

1. はじめに

偏微分方程式の代表的な数値解法として有限差分法・有限要素法・境界要素法の三つの手法がある[1]．有限差分法や有限要素法が対象とする領域全体を離散化して解を求めるのに対して，境界要素法は対象領域内の構成要素の境界部分のみを離散化することで解を求めるという特徴を持つ．この特徴により境界要素法は無限領域を容易に扱うことができ，またモデルの離散化に要する人的コストや問題の自由度を低く抑えることができる利点を持つ．一方，境界要素法では，一般的に離散化後に得られる連立一次方程式の係数行列が密行列となり，本方程式の求解に要する計算時間が多大となる傾向がある．そこで，実応用解析では，高速多重極法（FMM）[2] や H 行列法[3] といった高速な密行列ベクトル積演算手法を活用する場合が多いが，これらの諸技術に加えて分散・並列処理を利用することが大規模解析では不可欠となってきている．そこで本稿では，著者らが開発を行っている境界要素解析を支援するソフトウェアフレームワークについて，その全体構想と一般の境界要素解析を考慮し，密行列演算による実装を行った場合の結果について述べる．

2. 境界要素法

境界要素法では，まず基礎方程式から境界上での積分方程式を導出する．その後，境界部分を要素に分割，すなわち離散化を行って境界積分方程式を導出し，適切な境界条件及び初期条件を適用することにより，連立一次方程式に帰着させる．この連立一次方程式を解くことにより，境界要素法による解を得ることができる．図 1 に境界要素解析の処理手順を示す．

境界要素解析では，対象とする物理場により基礎方程式や境界積分方程式が異なったものとなり得るが，離散化された境界上で積分計算を行った結果を元に連立一次方程式を作成し，それを解くという処理手順は共通である．そこで，著者らはこれらの共通部分をフレームワーク化し，それを分散・並列化することにより，多様な境界要素解析において利用可能な並列化フレームワークの開発を試みた．

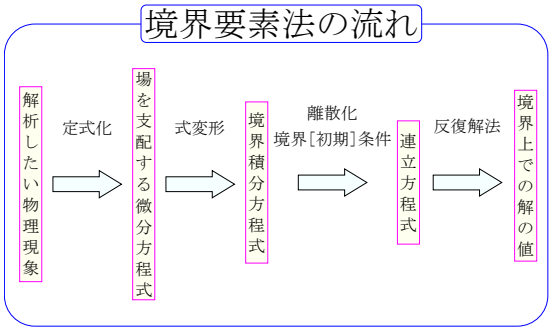

図 1 境界要素法の手順

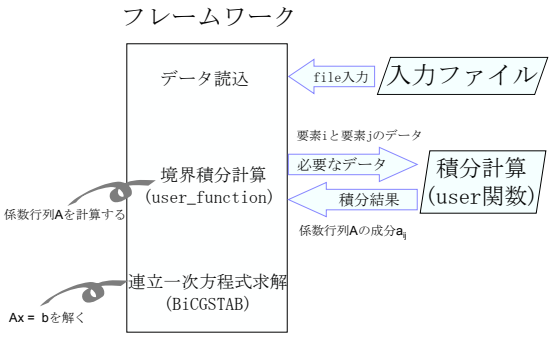

図 2 並列化境界要素解析フレームワーク

3. 並列化境界要素解析フレームワークの構成

開発する並列化フレームワークの構成を図 2 に示す．まず，モデルデータの入力部では，今回開発するフレームワークでは密行列演算を用いることから，行列の要素数はモデルの自由度の 2 乗となることを考慮して，マスタプロセスがファイルシステムからモデルデータをロードし，それを MPI_BCAST 関数で各プロ

† 京都大学
Kyoto University
†† 同志社大学
Doshisha University
††† JST CREST

セスに分配する方式を採用する．なお，今後開発を行う FMM や H 行列法を用いた実装では，各プロセスが分散化されたモデルデータを保持するため，MPI-IO を用いた実装を行う予定である．

各プロセスがモデルデータを読み込んだ後，連立一次方程式の係数行列を作成する．ここで，係数行列を求める積分計算は対象とする物理場の種別や離散化に採用する要素形状，積分の方式（解析積分／数値積分）によって多様性を持つ．そこで，本フレームワークでは，このような境界要素解析の多様性に対応するため，係数行列の各要素はユーザ自らが記述するユーザ関数により与える方式とした．すなわちフレームワーク側では，ユーザ定義関数の引数および返り値を定義し，係数行列作成に必要なモデルに関する情報をユーザ定義関数に引き渡す．係数行列作成部の並列化では，各プロセス・スレッドが同時並行的にユーザ定義関数を呼び出すことにより行う．なお，ユーザの中にはユーザ定義関数を含むより完全な形の境界要素解析プログラムの提供を望む場合があると想定される．そこで，そのような要求に応えるため，いくつかの典型的な解析に対してテンプレートという形でユーザ定義関数を提供する．具体的には，ラプラス場の境界値問題の一つである一次三角形要素を用いた表面電荷法のテンプレートを実装している．このテンプレートはユーザがユーザ定義関数を記述する際の参考となると考えられる．また，今後 FMM や H 行列法の実装を含むテンプレートを提供する予定であり，この場合テンプレート内の関数がフレームワーク側のいくつかの関数に代わって呼び出されることが行われる．なお，係数行列の右辺ベクトルについても係数行列と同様の形式で作成する．

最後に作成された連立一次方程式を BiCGStab 法[4] により解き，解ベクトルを得る．ここで，上記のユーザ関数を呼び出すループ文と BiCGStab 法の各ループは MPI と OpenMP によるハイブリッド並列化を行っている．

4. 性能評価

開発した並列化フレームワークと表面電荷法テンプレートを用いて，要素数 21600 の問題を T2K オープンスーパーコンピュータ（京大）上で解いた．

実行時間は逐次実行の場合，約 590 秒であったが，4 プロセス 4 スレッドのハイブリッド並列化を行った場合，約 40 秒に短縮された（図 3）．図 4 に本数値実験における台数効果を示す．16 プロセス 4 スレッドの場合で約 57 倍の台数効果が得られており，コア数に対して線形に近いほぼ理想的な速度向上が得られていることが分かる．

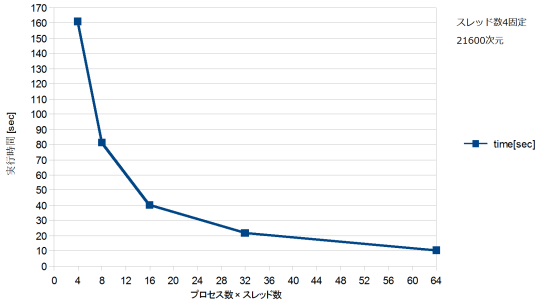

図 3 コア数に対する実行時間の変化

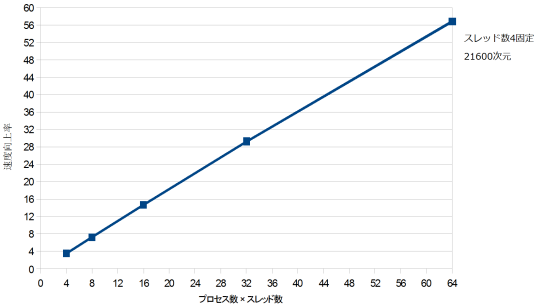

図 4 コア数に対する速度向上率の変化

5. まとめ

本稿では，並列化境界要素解析を支援するフレームワークの設計および MPI ／ OpenMP による実装を行った結果について述べた．21600 自由度の問題を開発したフレームワークによって解いた場合，逐次では約 590 秒であった実行時間が 4 プロセス 4 スレッドでは約 40 秒に短縮された．また，台数効果においてもほぼ理想的な効果が得られた．今後はより大規模なモデルに対する性能評価を行うとともに，FMM や H 行列法を活用した場合の対応について取り組んでいく予定である．

参考文献

1) 樫山和男, 牛島省, 西村直志 : 並列計算法入門, 計算力学レクチャーシリーズ, 丸善株式会社 (2003).
2) H. Cheng, L. Greengard, and V. Rokhlin, J. Comput. Phys. , vol.155, pp.468-498 (1999).
3) L. Grasedyck and W. Hackbusch, Computing, vol.70, pp.295-334 (2003).
4) Henk A. van der Vorst : Iterative Krylov Methods for Large Linear Systems, Cambridge University Press (2009).