領域分割法による境界要素解析の並列計算の効率の検討*

岩瀬 英仁^{*1}, 神谷 紀生^{*2}, 北 栄輔^{*3}Performance Evaluation of Parallel Boundary Element
Analysis by Domain Decomposition Method

Hidehito IWASE, Norio KAMIYA and Eisuke KITA

We previously applied on iterative scheme of domain decomposition algorithms for parallel computing to boundary element analysis in potential problems. Our aim of this work is to evaluate the efficiency of the scheme using a network of workstations, for different conditions of load balance in each domain. Application of complicated domain decomposition to a mold IC is considered as a example of industrial application.

Key Words: Boundary Element Method, Parallel Computing, Domain Decomposition Method, Cluster Computing, Workstations, Efficiency, 2D Potential Problem

1. 緒 言

著者らは、以前の研究⁽¹⁾において並列計算のための領域分割法による三つの反復計算アルゴリズムを境界要素解析に適用し、基本的な例題によって、それらの比較を行った。そこでわかったことは、これら分割法は収束性に違いがあるが、いずれのアルゴリズムも従来法に比べて、CPUのメモリー節約、プログラミングの容易化、データ作成の容易化に有効である。そこで、本研究では、並列計算のための領域分割法による境界要素解析の計算効率をさらに詳しく検討する。以前に構築したワークステーションによるクラスタ並列解析システムについて、各ワークステーションの負荷をいろいろに変化させてアルゴリズムの効率を評価する。さらに、多数の部分領域からなる二次元問題にアルゴリズムを適用して、領域分割法に基づく並列化計算法の実践的な問題に対する有効性を確認する。

2. 並列計算のための
領域分割境界要素解析

提案する方法のアルゴリズムについてはすでに文献⁽¹⁾で詳しく述べているので、ここでは後の議論に必要となる内容だけを簡単に示す。

解析対象として、二次元ポテンシャル問題を考える。境界要素解析では、支配微分方程式を積分方程式に変換し、これを境界要素に離散化して解を求める。そして、部分領域に分けて解析する場合、次のようになる。全体領域 Ω [図1(a)] を仮想境界(異種材料の複合体では材料ごとによって分けられる境界を指す)によって m 個の部分領域 $\Omega_i (i=1, 2, \dots, m)$ に分割し [図1(b)]、各部分領域について積分方程式を離散化すれば次式を得る。

$$\mathbf{G}^i \mathbf{q}^i = \mathbf{H}^i \mathbf{u}^i \quad (i=1, 2, \dots, m) \quad \dots\dots\dots (1)$$

ここで、 $\mathbf{u}^i, \mathbf{q}^i$ は仮想境界も含め、 Ω_i の境界の節点におけるポテンシャル u 、フラックス q のベクトル、 $\mathbf{H}^i, \mathbf{G}^i$ はそれらの係数行列である。なお、右肩の添字は領域の番号を表している。領域 Ω_k と Ω_l の間の仮想境界 Γ_{kl} 上ではポテンシャル値、フラックス値に関して、次の適合条件と平衡条件が成り立つ、

$$\mathbf{u}^k = \mathbf{u}^l, \mathbf{q}^k = -\mathbf{q}^l \quad \dots\dots\dots (2)$$

* 原稿受付 1995年12月28日。

^{*1} 正員、名古屋大学大学院 (〒464-01 名古屋市千種区不老町)。^{*2} 正員、名古屋大学情報文化学部。^{*3} 正員、名古屋大学大学院工学研究科。

ただし,

$$q^k \equiv \left(\mu \frac{\partial u}{\partial n} \right)^k \dots \dots \dots (3)$$

であり, n は境界での外向き単位法線ベクトル, μ は例えば熱伝導率などの物性値である.

通常の解析には式(1), (2)を連立して解く. これに対して並列計算では次のようなアルゴリズムを用いる. 本来, 未知量である仮想境界の境界条件を仮定して与え, 式(1)を各領域ごとに独立して解く. そして, この仮定した境界条件が収束するまで, 境界条件を更新しながら反復計算を行う. 仮想境界上の境界条件の更新方法としては, Uzawa 法⁽²⁾⁽³⁾, Schwarz のノイマン ノイマン法⁽⁴⁾, Schwarz のディリクレ ノイマン法⁽⁴⁾ などがあるが, 前報⁽¹⁾で示したように第三の方法が最も収束性がよかったので, 以下ではこの方法だけを採用する. 以下にこの方法を用いた並列境界要素解析の計算手順を示す.

Step 1: 仮想境界上の境界条件の初期値を設定する.

$$u^k = \bar{u}^k, q^l = \bar{q}^l \dots \dots \dots (4)$$

ここで上付きのバーは与えられた値であることを示している.

Step 2: 各部分領域において式(1)を解き, 未知量を求める.

Step 3: 収束判定を行う. すなわち, s 回めと $s-1$ 回めの計算値が許容値 ϵ_1, ϵ_2 に対して

$$|u_s^k - u_{s-1}^k| \leq \epsilon_1, |q_s^l - q_{s-1}^l| \leq \epsilon_2 \dots \dots \dots (5)$$

ならば計算を終了する. そうでなければ Step 4 に行く.

Step 4: Schwarz のディリクレ ノイマン法⁽⁴⁾により, 仮想境界上の境界条件を次式により更新する.

$$u_{s+1}^k = u_s^k + \gamma(u_s^l - u_s^k) \dots \dots \dots (6)$$

$$q_{s+1}^l = -q_s^k \dots \dots \dots (7)$$

更新終了後, Step 2に戻る. なお, 式(6)の γ は定数であり, 利用者が設定する.

(a) Single domain (b) Multiple subdomains

Fig. 1 Domain under consideration

3. クラスタコンピューティング システムの計算効率の検討

3章では, 前報⁽¹⁾で構築したクラスタコンピューティングを用いて, 領域分割法による境界要素ポテンシャル解析の並列計算の効率について考察する. ワークステーションの機種は Fujitsu S 4/EC であり, これらは転送速度 10 Mbps のイーサネットケーブルで結合されている. 本研究で用いた各ワークステーションの仕様は同じであり, 並列計算での各ワークステーション間の通信方式は UDP/IP を採用した.

取り上げた例題は, 正方形領域の二次元ポテンシャル問題で, 全体領域を四つの部分領域に分割する. そして, (1)各部分領域の要素数を均等として, 全体要素数を変化させた場合, (2)各部分領域の要素数を不均等として, 負荷量のアンバランスを変化させた場合, の二つの場合について並列計算の効率を調べる. システム構成はホストマシンが1台, サブマシンは4台である. 各ワークステーションの処理内容は次のようである. ホストマシンは計算全体の制御とともに, 仮想境界上の境界条件の更新を行い, 各サブマシンは係数行列の計算と連立一次方程式の計算(LU分解, 代入計算)を行う. 係数行列の計算とLU分解は最初に一度だけ行い, その結果得られた行列は保存され, 反復計算で繰返し使われる.

なお, 以下の計算には, 境界要素として一定要素を採用した. 解の収束判定は仮想境界のすべての点で更新最大値が $\epsilon_1 = \epsilon_2 = 10^{-3}$ 以下になったときに反復計算終了とした.

3.1 各サブマシンの負荷量が同じで, 負荷量を変化させた場合 例題の境界条件, 形状寸法を図3に示す. 領域分割は, 全体正方形領域を1辺の長さが50の正方形部分領域に分ける. 各部分領域の要素数を24(1辺当たり6要素), 60(1辺当たり15要素), 120(1辺当たり30要素)の場合について解析を行い, 並列計算の効率を調べる. 各部分領域の仮想境界条件の初期値はポテンシャル, フラックスとも零とし, γ は各場

Fig. 2 Domain under consideration Ω_k and adjacent domain Ω_l

合とも0.2とした。なお、並列計算の効率は1台のワークステーションによる直列計算時間と、クラスタコンピューティングによる並列計算時間を比較し、算出した。その結果を表1に示す。各ケースを1台のワークステーションで直列計算を行った場合の計算時間の内訳を図4に、並列計算でのホストマシンの処理時間の内訳を図5に示す。直列計算の計算時間は係数行列の計算、LU分解、更新計算の時間すべてを含んでいる。図より計算時間の増加は要素数の増加率の二乗に比例することがわかる。また、図5において、ホストマシンの待機時間はサブマシンの計算結果の待ち時間である。ケース1では同期待ち時間と通信時間が並列処理による計算量の減少分を相殺し、ほとんど計算効率は改善されていない。これに対して、ケース2、3では50%以上の効率改善が確認できた。各ケースとも反復計算の回数は25回であり、誤差の許容値をさらに小さく取ると、反復計算の回数は増加し、計算効率が少し低下してくる。通信時間はデータの転送時間とオーバーヘッドからなり、データの転送時間は各ケースのおよそのデータ量(192 byte, 480 byte, 960 byte)をイーサネットケーブルの転送速度10 Mbyte/sで割ったものとして計算すると無視できるほど小さく、通信時間のほとんどがオーバーヘッドであることがわかる。

3.2 各サブマシンの負荷量が異なり、負荷量のアンバランスさを変化させた場合 各部分領域の要素数が異なる(すなわち各サブマシン間の負荷量がアン

バランスな状態)場合について、アンバランスさの変化に対する並列計算の効率の変化を調べる。用いる例題は図3と同じであり、図6に示すように四とおりの場合を考え、それぞれについての計算効率を調べる。ケース1では各部分領域の大きさは同じであり、各領域の要素数を100とする。したがって、各領域の計算負荷量は同じになる。ケース2から4では、領域 Ω_2 と領域 Ω_3 の要素数は100のままとして、 Ω_1 を小さくするようにして領域 Ω_1 と領域 Ω_4 間で計算負荷量のアンバランスな状況を設定する。各ケースにおける各部分領域の要素数を表2に示す。ここで、各ケースでの負荷のアンバランスさを表すものとしてアンバランス指標を考え、この指標は部分領域の最大要素数と最小要素数の比で定義する。なお、定数 γ は0.3、仮想境界条件の初期値は零として計算した。各部分領域 $\Omega_1, \dots, \Omega_4$ の解析はサブマシン1から4に担当させる。各ケースについてクラスタコンピューティングによる並列計算で改善された計算効率は、1台のワークステーションによる直列計算時間と比較して、算出した。その結果を図7に示す。ケース1は各領域の要素数が同じ場合で、改善された計算効率は70%で最も高く、ケース2では54%、ケース3では46%、ケース4では32%と

Table 1 Improved efficiency in each case

	Number of elements in one domain	Improved efficiency (%)
Case 1	24	3
Case 2	60	56
Case 3	120	70

Fig. 3 Four domains decomposition problem

Fig. 4 Serial computing time in each case

Fig. 5 Parallel computing time at host machine in each case

順番に計算効率が低くなった。アンバランス指標が大きいほど、すなわち、各部分領域の要素数で最大のものとの最小のものとの開きが大きいほど計算効率が低下している。これは最大要素数が大きくなると、その部分領域での解析時間が長くなって、ホストマシンの同期待ち時間が長くなり、計算効率が低下するためである。その詳細を検討するため図8にケース1とケース4での各ワークステーションのタイムチャートを比較して示す。なお、図中の仮想境界条件更新の計算時間の内訳は、ホストマシン側では仮想境界条件の更新と同期待ち時間、サブマシン側では代入計算と同期待ち

時間が含まれている。

この結果の傾向は、並列計算を行うときにあらかじめ予想されたことであり、これを実際例について定量的に明らかにし、負荷にアンバランスがあるときには注意すべきことを改めて確認したものとなる。なおアンバランスがあるときには最も負荷の大きいサブマシンによって計算効率が制約される。実用上各サブマシンの負荷を同じにすることは困難なので、負荷のばらつきを可能な限り小さくすることが重要である。

4. 多数領域問題への応用

4.1 一様材質場の9部分領域分割 図9に示すように正方形領域を九つの等しい大きさの正方形部分領域に分割する。計算方法は前に述べたものと同じとし、定数 γ は0.2とした。境界条件、仮想境界条件は図9に示すように与え、初期値は零を与えた。この例題を並列計算するにはホストマシン1台とサブマシン9台が必要で、各部分領域 Ω_1 から Ω_9 の解析をサブマ

Table 2 Number of elements in each case

	Ω_1	Ω_2	Ω_3	Ω_4	Indicator of load unbalance
Case 1	100	100	100	100	1.0
Case 2	80	100	100	120	1.5
Case 3	60	100	100	140	2.3
Case 4	40	100	100	160	4.0

Fig. 6 Unbalance load in domain decomposition

Fig. 7 Improved efficiency of four domains decomposition problem in cluster computation

Fig. 8 Time chart of parallel computing in Case 1 and Case 4

シン1から9に担当させる、並列計算による計算効率は、1台のワークステーションによる直列計算時間と比較して算出した。境界要素分割は各部分領域の1辺を6要素として、一つの部分領域の要素数を24要素とした場合、1辺を15要素として、一つの部分領域の要素数を60要素とした場合について解析を行った。先と同じように解析には一定要素を用いた。図中の u, q は仮想境界で仮定する変数を示している。

24要素の場合の解の収束の結果を図10に示す。図中の point 1 から point 6 は図9の仮想境界上の白丸点を左から右に順番に対応させてある。その結果、各部分領域が24要素の場合には計算効率はほとんど改善されず、ホストマシンの計算時間の内、通信時間と同期待ち時間が全体時間の90%を占め、並列処理による計算量の減少分が相殺された。1台における計算時間と通信時間が同じオーダーなので通信時間が計算時間に大きく影響し、これが計算効率が改善されない理由である。これに対して、各部分領域が60要素の場合には計算効率は64%改善された。計算時間は要素数の増加率に対してほぼ2乗に比例し、60要素の計算時間は24要素の計算時間よりも5倍近くになる。通信時間は要素数の増加に対して比例して増大し、また同期待ちの

時間は部分領域の数に比例して増大する。60要素の場合には並列処理全体の時間における通信時間の占める割合が小さくなり、並列処理による計算量の減少分が計算効率の改善となって現れた。このことから、部分領域数が同じであるとき、各部分領域の要素数が増えれば、計算効率が改善されと考えられる。3章における4部分領域と4・1節における9部分領域の場合の結果からわかることは、並列計算での計算効率は、各部分領域における未知数の数が密接に関連することである。

4・2 モールド IC の温度解析 ここでは、複雑な構造の例としてモールド IC の温度分布を領域分割法で解析する。モールド IC の境界要素法熱伝導・熱応力解析は佐藤ら⁽⁵⁾が従来法による領域分割法で解析しており、ここではその構造を参考にモデル解析を行った。モールド IC とは、IC チップとそれを置くパッドおよびリードをモールド樹脂で固めたものであり、その構造は複雑かつ微細であり、それをモデル化したものを図11に示す。図中の IC チップ、樹脂、パッド(リードを含む)それぞれの物性値の著しい違い(表3)がパッケージの割れ、断線や誤動作を引き起こす原因となっている。ここでは問題を単純化し、IC による内部発熱は考えないで、IC の境界条件は温度 $u=100^{\circ}\text{C}$ を与え、領域の支配微分方程式は二次元ラプラス方程式として解析を行う。まず最初に、図12に示すようにモールド IC のモデル構造を八つの部分領域に分割する。各部分領域を長方形にしたのは形状データの作成を容易にするためである。実境界条件は周囲断熱条件

Fig. 9 Nine domains decomposition problem

Fig. 10 Convergence of solution in nine domains decomposition problem

	Heat conductivity ($\text{W/m} \cdot \text{K}^{-1}$)
IC chip	84.0
Resin	0.6
Lead	60.0
Pad	60.0

Fig. 11 Model of mold IC in temperature analysis

Fig. 12 Eight domains decomposition

とし、その他は図12に示すように与えた。仮想境界条件の更新方法は先と同じであり、仮想境界条件の初期値は零とした。また、先と同じように一定要素を用いた。解の収束の結果を図13に示す。図中のa, b, cは図12のa, b, cの位置に対応する。モールドICの代表点での温度分布を図14に示す。この例題を並列計算するのに、ホストマシン1台とサブマシン8台を準備し、各部分領域 Ω_1 から Ω_8 をサブマシン1から8までに担当させる。並列計算による計算効率は、1台のワークステーションによる計算時間と比較して算出すると47%改善された。この問題を、従来法の領域分割境界要素法で解析するには、内部境界が複雑なため全体方程式を構成することは大変面倒であるのに対し、本方法では非常に容易である。

5. 結 言

ワークステーションによるクラスタコンピューティング環境での領域分割法による境界要素解析の並列計算の効率を検討した。各部分領域の要素数が同じ場合、すなわち、各サブマシンの計算負荷量と同じになる場合は要素数が多くなるほど計算効率が上がることがわかった。また、各領域の要素数が異なる場合、すなわち、各サブマシンの計算負荷量に差がある場合は、最大要素数と他の領域との要素数の差が大きくなるほど同期待ち時間が長くなり、計算効率が低下することが再確認された。実用上、各領域の要素数を完全に同じにすることは難しいので、各領域の要素数のばらつきを可能な限り小さくすることが重要である。

さらに、多数領域問題では、一様材質の9部分領域問題と、モールドICの温度解析で複数の材質からな

Fig. 13 Convergence of solution in mold IC

Fig. 14 Temperature distribution in model of mold IC

る8部分領域問題を取り上げた。その結果、精度よい結果が得られるとともに、また、クラスタコンピューティングにより計算時間の大幅な短縮がなされ、並列計算のための領域分割法は計算効率改善に極めて有効な方法であることが確認された。

今後は他の問題への適用、他の並列方式による計算効率も検討してゆきたい。

文 献

- (1) 岩瀬英仁・神谷紀生・北栄輔, 領域分割法による境界要素解析の並列計算のためのアルゴリズムの検討: 解の収束性, シミュレーション学会論文集, **15** 3 (1996), 掲載予定.
- (2) Glowinski, R., Dinh, Q.V. and Periaux, J., Domain Decomposition Methods for Nonlinear Problems in Fluid Dynamics, *Comput. Meth. Appl. Mech. Eng.*, **40** (1983), 27-109.
- (3) 矢川元基・曾根田直樹, パラレルコンピューティング, (1991), 培風館.
- (4) Lai, C.H., Diakoptics, Domain Decomposition and Parallel Computing, *Comput. J.*, **37** (1994), 840-846.
- (5) 佐藤満・結城良治・松本敏郎, モールドICの境界要素法熱伝導・熱応力解析, 境界要素法論文集, **3** (1986), 219-224.