領域分割形並列接触アルゴリズムの開発と
動的陽解法解析コードへの実装*大石 篤哉^{*1}, 吉村 忍^{*2}, 矢川 元基^{*2}Domain Decomposition Based Parallel Contact Algorithm
and Its Implementation to Explicit Finite Element Analysis CodeAtsuya OISHI^{*3}, Shinobu YOSHIMURA and Genki YAGAWA^{*3} The University of Tokushima, Dept. of Mechanical Engineering,
2-1 Minami-johsanjima-cho, Tokushima, Tokushima, 770-8506 Japan

Computer simulations are about to replace experiments in various fields related to dynamic problems, such as crashworthiness of vehicles, vibration of a nuclear reactor in earthquake and ultrasonic wave propagation in solid with cracks for nondestructive evaluation. For solving such large scale problems, the parallel processing has become a key technology. In this paper, a parallel contact algorithm suitable for a domain decomposition technique is proposed, and implemented on a dynamic finite element analysis code based on an explicit time integration scheme. Parallel efficiency of the code is tested through sample analyses on a PC cluster.

Key Words: Parallel Computing, Finite Element Method, Contact Impact, Dynamic Analysis, Domain Decomposition

1. 緒 言

近年の計算機性能の飛躍的向上により、従来模型や実機を用いていた実験を有限要素法などの数値解析手法によるシミュレーションに置き換える動きが盛んになっている。このため、要求される解析規模は急速に増大しており、大規模な問題を許容しうる速さで解くために同時に複数の計算機を用いて処理を行う並列処理が用いられるようになってきている⁽¹⁾。並列処理環境としては、分散メモリ形専用並列計算機、数個のCPUを持つ共有メモリ形マルチプロセッサ機などのほかに、数台から数百台のパーソナルコンピュータをネットワークで結合したPCクラスタなどがある。並列処理環境を有効に利用し高速な処理性能を得るためには、並列処理に適したアルゴリズムが必要となるが、従来の有限要素解析ソフトウェアは並列処理を想定していないため、並列処理環境において良い性能を得ることが難しい。このため並列処理環境に適した新しい有限要素解析ソフトウェアの開発が急務となってい

る。

自動車の衝突解析や物体の落下時の解析などに必須となる接触解析⁽²⁾⁽³⁾についても並列化が試みられてきた。接触解析では二つの物体(あるいは一つの物体のうち二つの部分)がある時刻に接触するときの接触位置および相互作用を解析する。物体同士の接触は次のように定義される。

$$\Omega_1 \cap \Omega_2 = \phi \cdots \cdots (1)$$

$$\Gamma_1 \cap \Gamma_2 \neq \phi \cdots \cdots (2)$$

ここで、 Ω_1, Ω_2 は物体、 Γ_1, Γ_2 はその表面を示す。式(1)は物体間の貫入がないことを示している。

代表的な接触解析アルゴリズムとしては、あらかじめ接触する可能性のある面について一方を master, 他方を slave とみなして探索を行う master-slave algorithm⁽⁴⁾ (汎用コード DYNA3D に採用されている)、幾何モデルの階層 [面, セグメント (表面に露出した要素の 1 面) 等] ごとにそれを包含し座標軸に平行な直方体 (territory と呼ばれる) を構成し階層的に接触探索範囲を絞り込む HITA⁽⁵⁾⁽⁶⁾ (Hierarchical Territory Algorithm)、表面の要素を球面で表現し近接する二つの球の半径の和と中心間距離との比較により接触を判定する pinball algorithm⁽⁷⁾⁽⁸⁾ などがある。一般に、接触解析は通常の線形解析や非線形 (弾塑性)

* 原稿受付 1999 年 12 月 17 日。

^{*1} 正員, 徳島大学工学部 (☎ 770-8506 徳島市南常三島町 2-1)。^{*2} 正員, 東京大学大学院 (☎ 113-8656 東京都文京区本郷 7-3-1)。

E-mail: oishi@me.tokushima-u.ac.jp

解析と比べて並列化が難しい。接触解析では、物体表面の節点同士または節点とセグメント間の距離を計算し接触が起きているか否かを判定する接触探索(contact search)が必須であるが、接触アルゴリズムを並列化する場合には、接触する節点、要素のペアを探す接触探索プロセスの並列化が最大の課題となる。一般に接触を起こす節点、要素はあらかじめ指定されることはなく、また解析の進行に伴い新たな接触が生じたり、接触していた部分が離れるなどつねに状態が変化する。このため、接触点を探索する範囲が広くなり、解析規模の増大とともに探索に要する時間が増大する。特に大規模解析のプラットホームとして主流となる分散メモリ形並列処理環境においては、解析領域のデータが複数のプロセッサに分散して管理されるため、探索時のプロセッサ間通信によるオーバーヘッドが効率低下の要因となる。

本研究では、領域分割形並列処理に適した並列接触アルゴリズムを開発し、領域分割法に基づく非定常並列有限要素解析コードに本アルゴリズムを実装するとともに例題によりその並列処理性能を検証した。

2. 非定常問題の領域分割形 並列有限要素解析

2.1 基本原理：領域分割法 非定常問題の有限要素式は、減衰を考えない場合次式で与えられる。

$$[M]\{\ddot{u}\} + [K]\{u\} = \{f\} \quad \dots\dots\dots (3)$$

ここで $[M]$, $[K]$, $\{\ddot{u}\}$, $\{u\}$, $\{f\}$ はそれぞれ質量マトリックス、剛性マトリックス、加速度ベクトル、変位ベ

クトル、荷重ベクトルである。

有限要素法の並列処理方式としては、(1)節点単位の並列処理、(2)要素単位の並列処理、(3)部分領域単位の並列処理という三とおりが考えられる。このうち、領域単位の並列化は並列処理の単位(粒度)を大きくとれるためさまざまな並列処理環境で効率的な解析が可能になる。

領域分割法⁽⁹⁾⁽¹⁰⁾は解析領域を複数の部分領域に分割して解析する並列処理アルゴリズムであり、領域分割法による非定常問題の解析プロセス⁽¹¹⁾⁽¹²⁾は下記のようにまとめられる。

- (1) 全体領域を複数の部分領域に分割する。
- (2) 時間ステップ $n=0$
- (3) 部分領域間境界の変位の初期値(任意)を与える。
- (4) 各部分領域において有限要素解析を行う(並列化可能)。
- (5) 部分領域間境界での反力の不釣り合いに基づき領域間境界の変位を更新する。
- (6) 領域間境界の変位が収束していなければ(4)へ。
- (7) 時間ステップ $n=n+1$
- (8) 残り(時間)ステップがあれば(3)へ。
- (9) 解析終了。

(5)における領域境界の変位の更新には共役こう配法を用いている。本アルゴリズムは陰解法、陽解法ともに適用可能であるが陽解法では(4)～(6)の反復が不要となる。

図1にフローチャートを示す。最内周の部分領域に関するループが並列化可能である。

2.2 並列処理環境への実装 領域分割法では各部分領域における有限要素解析は並列処理可能であり、さまざまな並列処理環境に実装可能である。複数のプロセッサを用いて並列処理を行う場合、同一プロセッサまたは異なるプロセッサ上のプロセス間でデータのやりとりが必要となる。異なるプロセス間でデータを

Fig. 1 DDM flowchart

Fig. 2 DDM Implementation to WS cluster

やりとりする「メッセージ・パッシング」にはさまざまな方法があるが、標準的なライブラリーとしてPVM(Parallel Virtual Machine)⁽¹³⁾やMPI(Message Passing Interface)⁽¹⁴⁾が開発され広く用いられている。

図2に、本研究の領域分割形並列有限要素解析コードのワークステーションクラスタへの実装図を示した⁽¹²⁾。本研究の領域分割法による並列有限要素解析システムは、1個のコントローラと複数のアナライザから構成される。このうち、アナライザは各部分領域の解析を行うプログラムであり、各アナライザは一つの部分領域の有限要素解析を行う。コントローラは、(1)アナライザへの解析用データの割り当て、(2)部分領域間境界節点でのデータ修正、などの仕事を行う。1個のプロセッサにコントローラを置き、残りのプロセッサにアナライザを配置する。部分領域数よりプロセッサ数が少ない場合には、1個のプロセッサが複数の部分領域を担当することになる。この場合、pool-of-tasks方式⁽¹⁵⁾の動的負荷分散によりプロセッサの使用効率を向上させる。

3. 領域分割形並列接触アルゴリズム

3.1 接触データの配置 接触解析アルゴリズムとしては、あらかじめ接触しそうな面(接触面)の対について一方をmaster、他方をslaveとみなして探索を行うmaster-slave algorithm⁽⁴⁾に準じ、領域分割形並列化を行った。master-slave algorithmの領域分割形並列化としては、ある一対の接触面に関して、接触面全体を一つの部分領域に割付ける方式が考えられる⁽¹⁶⁾⁽¹⁷⁾。しかしながら、接触面が広い場合には、その接触面全体を含む極端に大きな部分領域ができるため、プロセッサ間の負荷分散が困難になるという欠点がある。また、領域分割の際に接触面の部分に特別な制約条件を付加して分割を抑制するなどの制限も必要となる。このため、この手法の適用は接触面の部分が比較的小さい場合に限定される。

したがって、さまざまな場合に柔軟に対応できるためには、接触面自体を複数の領域に分割することが必須となる。一つの接触面を複数の領域に分割する場合、接触に関与するデータの各プロセッサへの配置として下記のような方法が考えられる。

- (1) 接触面全体のデータを当該接触面に関与する領域すべてに配置する
- (2) 各領域には当該接触面のデータ中で必要な部分のみ配置する
- (3) 接触面の分割は変形解析のための通常の領域

分割とは別の分割を行い、データ配置も通常の領域分割とは独立させる。

方法(1)では、single surface contact⁽¹⁸⁾など接触面が大きい場合に送信データサイズの増大が問題となる。方法(3)の例としては、Hooverら⁽¹⁷⁾、Attawayら⁽¹⁹⁾の研究があるが、異なる分割パターンに起因するデータ管理・通信パターンの複雑化を伴う。方法(2)では、各部分領域にそれぞれ異なるデータを用意する必要があるものの、各部分領域に渡す接触解析用データサイズはコンパクトになるという利点がある。この方法の研究例としては、Malone and Johnson⁽²⁰⁾、Elsnerら⁽²¹⁾、Plaskacz⁽²²⁾などがある。Malone and Johnson および Plaskacz は、部分領域とプロセッサを1対1に対応させ、それぞれ“Volume”(シェル要素を包含する最小直方体)、“cell”(各節点が属する等間隔の三次元格子)の包含関係により、接触解析に必要なデータに対応するプロセッサから送信させている。しかし、特に解析中に計算負荷が変化する接触解析においては、部分領域とプロセッサを静的に1対1対応させることは負荷の不均衡を招き効率が低下する要因となる。これに対し、Elsnerらは、静的な配置ではあるが一つのプロセッサに複数の部分領域を担当させることで負荷の不均衡を低減させている。また、各領域に属する接触面を相互の距離により階層化し探索範囲を絞っているが、16 CPU 使用時に4倍程度のスピードアップにとどまっている。

本研究では、領域分割の自由度が高く、多様な条件に対応可能でスケーラビリティを得やすく、また各種並列処理環境にも適応しやすい方法(2)を採用した。特に、実装対象である2.2節の動的有限要素解析コードの持つ「静的な負荷分散だけではなく、完全に動的な負荷分散にも対応できる」という特徴を維持することにした。ここで、完全な動的負荷分散とは、部分領域とプロセッサの対応が固定されておらず、しかも部分領域数がプロセッサ数よりも十分多いため解析領域全体のデータが同時にプロセッサ群に載ることがない状態を意味する。これにより、静的な負荷分散と比べて計算速度は劣るものの、使用する計算機の容量以上の解析も可能になるという利点がある。なお、計算機容量内の解析には、静的な負荷分散(ただし各プロセッサに複数の部分領域を配置)を行う。

図3に本研究の領域分割形並列接触アルゴリズムにおける接触面のデータ配置例を示した。slave側接触面を構成する部分領域S2担当のプロセッサには、master側接触面のデータのうち部分領域S2と近接するM1-M3部分領域に含まれるmaster側接触面

のデータのみが併せて配置される。

3・2 領域分割形接触探索 本研究の並列接触アルゴリズムでは、下記のような六つの Part からなる領域分割形接触探索を行う。なお、拘束条件にはペナルティー法を用いている⁽³⁾。また、簡単のために摩擦のない場合に関して述べるが、摩擦ありの場合への拡張も可能である。

Part I :

本研究の領域分割形並列接触アルゴリズムにおける領域分割形接触探索のための接触面部のデータ配置について説明する。接触面には、順に 0, 1, 2, ... と番号をつけ、接触面 0, 接触面 1, 接触面 3, ... と呼ぶ。簡単のために接触は 0 番と 1 番, 2 番と 3 番のように決められた対同士のみを考慮し, 1 番と 2 番, 1 番と 3 番, あるいは 2 番同士は接触しないとする。このように定義した接触面の対のうち一方を master, 他方を slave と呼ぶ。接触面 n の内部分領域 i に含まれる部分を構成する節点の集合を S_i^n と表す。 S_i^n を構成する節点について、その座標の最小値, 最大値をそれぞれ $X_{\min}, X_{\max}, Y_{\min}, Y_{\max}, Z_{\min}, Z_{\max}$ とすると、図 4 のように $(X_{\min}, Y_{\min}, Z_{\min})$ および $(X_{\max}, Y_{\max}, Z_{\max})$ を頂点とし S_i^n を包含する直方体 V_i^n (各辺は座標軸に平行) が定義される。さらに、 V_i^n を x, y, z 方向に d だ

け膨張させ $(X_{\min}-d, Y_{\min}-d, Z_{\min}-d)$ および $(X_{\max}+d, Y_{\max}+d, Z_{\max}+d)$ を頂点とする直方体を $\bar{V}_i^n$ とする。

Part II :

本研究の並列接触アルゴリズムでは上記で定義した直方体をもとに接触探索範囲の第一段階の絞り込みを行う。距離計算によらず包含関係だけで絞り込みを行うが、Part I で V の膨張処理を行っているため、各時間ステップごとに Part を実行することは不要となる。

接触面 n は接触面 $m[m=n+(-1)^n]$ と対となるため、 S_i^n に関する接触探索は接触面 m のうち S_j^m との距離が十分に接近している部分に対して行えば良いことがわかる。すなわち、slave 側節点 S_i^n に関する master 側接触探索範囲は $S_j^m(j \in J)$ となる。ここで、 J は次式で定義される。

$$J = \{j | \bar{V}_i^n \cap \bar{V}_j^m \neq \emptyset\} \dots\dots\dots (4)$$

Part III :

Part II の絞り込みに基づいて、slave 側節点 S_i^n に関する接触探索を行うプロセッサには S_i^n のデータと併せて $S_j^m(j \in J)$ に関するデータを配置する。すなわち、 S_i^n に関する接触探索を行うプロセッサには S_i^n と $S_j^m(j \in J)$ が配置される。ここで、

$$T_i^n = S_i^n \cup \left(\bigcup_{j \in J} S_j^m \right) \dots\dots\dots (5)$$

とし、 T_i^n に含まれる節点を下記のように bucket (各座標軸に平行な辺で構成された直方体群) に分割する。

- (1) T_i^n に含まれる節点の座標値について最小値および最大値を求め、それぞれ $X_{\min}, X_{\max}, Y_{\min}, Y_{\max}, Z_{\min}, Z_{\max}$ とする。
- (2) x, y, z 各方向の bucket サイズをそれぞれ bs_x, bs_y, bs_z とおく。
- (3) T_i^n の各節点 $P(x, y, z)$ は、それぞれ bucket

Fig. 3 Decomposition of contact surfaces

Fig. 4 Parallelepiped including a contact surface

番号 (bn_x, bn_y, bn_z) で指定される bucket に属する。
ここで, bn_x は下記の式により与えられる。

$$bn_x = (\text{int}) \frac{(x - X_{\min})}{bs_x} \dots\dots\dots (6)$$

(int) a は a を超えない最大の整数を意味している。
 bn_y, bn_z も同様に定義される。

Part IV :

S_i^j 上のある節点に関して, 最短距離にある $S_j^m (j \in J)$ 上の節点を探索する。このとき, S_i^j 上の節点が bucket 番号 (bn_x, bn_y, bn_z) で表される bucket に属すれば, その節点に関する接触探索範囲は, $S_j^m (j \in J)$ のうち隣接する 27 個の bucket に含まれる節点だけに限定される。

Part V :

slave 側節点 P_s の master 側における最近接節点 P_m が決まると, ここからは node-segment 形接触探索⁽⁴⁾と同様の局所的接触探索を行う。以下にその概要を示す。なお, 要素は六面体 8 節点一次要素, セグメント (segment) は四角形 4 節点としている。ここで, セグメントとは接触面を構成する各要素の面を指し, 六面体要素では各面が四角形であるためセグメントも四角形 4 節点となる。シェル要素については表裏それぞれ一つのセグメントとみなすことにより本アルゴリズムを適用可能である。

S_i^j 上の節点 P_s に対して最短距離にある $S_j^m (j \in J)$ 上の節点 P_m が見つかり, 次はその節点 (P_m) を含む各セグメントに対して接触探索を行う。図 5 に示すように, P_m を起点として, セグメントを構成する節点のうち反時計回りに見たときの隣接節点を終点とするベクトルを $\vec{c}_1$, 時間回り側のベクトルを $\vec{c}_2$, P_s を終点とするベクトルを $\vec{r}$, $\vec{r}$ のセグメント面への射影ベクトルを $\vec{r}$ とする。このとき下記不等式を満たすセグメントに対してのみさらに詳細な接触探索を続行すればよい。

$$(\vec{c}_1 \times \vec{r}) \cdot (\vec{c}_1 \times \vec{c}_2) > 0 \dots\dots\dots (7)$$

$$(\vec{c}_1 \times \vec{r}) \cdot (\vec{r} \times \vec{c}_2) > 0 \dots\dots\dots (8)$$

ここで,

$$\vec{r} = \vec{t} - (\vec{t} \cdot \vec{m}) \vec{m} \dots\dots\dots (9)$$

Fig. 5 Localized contact search

$$\vec{m} = \frac{\vec{c}_1 \times \vec{c}_2}{|\vec{c}_1 \times \vec{c}_2|} \dots\dots\dots (10)$$

である。

条件式 (7), (8) を満足するセグメントにおいて, セグメント面上の接触点の局所座標 (ξ_c, η_c) は次式を Newton 法で解くことにより求めることができる。

$$\frac{\partial \vec{r}}{\partial \xi} (\xi_c, \eta_c) \cdot \{\vec{t} - \vec{r}(\xi_c, \eta_c)\} = 0 \dots\dots\dots (11)$$

$$\frac{\partial \vec{r}}{\partial \eta} (\xi_c, \eta_c) \cdot \{\vec{t} - \vec{r}(\xi_c, \eta_c)\} = 0 \dots\dots\dots (12)$$

さらに侵入深さ g は次式で求められる。

$$g = \vec{m} \cdot (\vec{t} - \vec{r}(\xi_c, \eta_c)) \dots\dots\dots (13)$$

ただし, g が負の値の場合のみ接触が起きていると判定される。

実際に接触が起きている場合, slave 節点および master セグメント上の各節点には, 次式によりそれぞれ $\vec{f}_s, \vec{f}_m^i (i=1 \sim 4)$ という接触力が与えられる。

$$\vec{f}_s = -g\alpha\vec{m} \dots\dots\dots (14)$$

$$\vec{f}_m^i = \phi_i(\xi_c, \eta_c) \vec{f}_s \dots\dots\dots (15)$$

ここで, α は各セグメントごとに形状および特性値により決まる係数, $\phi_i(\xi, \eta)$ は形状関数であり, $i (i=1 \sim 4)$ はセグメントを構成する各節点に対応する。

なお, node-segment アルゴリズムは特定の接触状態のとき解が求められないことがあり, これを解消する inside-outside アルゴリズム⁽²³⁾が提案されている。

Part VI :

Part V で求められた接触力 (等価節点接触力) の集計を行う。各領域からの寄与を集計した正しい接触力は次式で表される。

$$\vec{f}_c = \vec{f}_s + \sum \vec{f}_m \dots\dots\dots (16)$$

ここで, $\vec{f}_c$ は集計後の接触力, $\vec{f}_s$ はその節点を slave 節点としたときの接触力, $\vec{f}_m$ はその節点が master 側節点として相対する接触面から受ける接触力であり, $\sum$ は複数の領域からの寄与を考慮することを示している。

3.3 並列処理環境への実装

2.1 節で述べた並列有限要素解析システムへの本アルゴリズムの実装方法は以下になる。Part I は各部分領域ごとにアナライザ側で並列処理を行う。Part II はコントローラで逐次的に処理を行う。なお, 式 (4) の演算負荷が過大になる場合は複数のアナライザによる分散処理も可能である。Part III は各アナライザで並列処理される。なお Part I ~ Part III は必ずしも毎時間ステップ行う必要はない。Part I において V の膨張処理に用いたパラメータ d や Part III における bucket サイズを適切な大きさにとり, 変形が大きい場合は短い時

間隔で、変形が小さい場合は長い時間間隔で再計算すれば良い。Part IVおよびPart Vは各アナライザで並列処理される。Part VIはコントローラで逐次処理される。

4. 並列接触アルゴリズムの 動的陽解法解析コードへの実装

ペナルティ法に基づく接触解析における有限要素法は次式となる。

$$[M]\{\ddot{u}\} + [K]\{u\} = \{f\} + \{f_c\} \dots\dots\dots (17)$$

ここで $\{f_c\}$ は接触力を表している。

2章で述べた領域分割形並列有限要素解析コードのうち時間積分法に中心差分法を用いた動的陽解法有限要素解析コードに本研究の領域分割形並列接触アルゴリズムを実装し、動的陽解法領域分割形並列接触解析コードを開発した。本コードの開発においては2章で述べた領域分割形並列有限要素解析法の特徴である下記の3項目の性質を損なわないように配慮した。

- (1) 部分領域数は使用するプロセッサ数の数倍から数十倍以上。
- (2) 全部分領域を一度にプロセッサの主記憶領域上に配置することはできない。
- (3) 領域分割時の制約条件はできるかぎり少なくする。

条件(1)および(2)に対して、本コードは pool-of-tasks 方式の動的負荷分散機構を用いることにより各プロセッサを効率的に使用でき、高い並列処理効率を得ることができる。

また、結果として使用する並列処理環境の容量以上の解析が実行可能であるという特徴を保っている。条件(3)に関しては、本コードで用いた領域分割形並列接触アルゴリズムが自由な領域分割に対応しており、領域分割時の制約は非常に少ない。

条件(2)に関して、接触解析においては接触力を相手側の部分領域に伝える必要があるため、各部分領域で評価した接触力をコントローラにおいて集中管理を行うことになる。このため、本研究の動的陽解法領域分割形並列接触解析コードでは、全部分領域を(1)接触面を有する部分領域(D1)、(b)接触面を持たない部分領域(D2)に区分し、各時間ステップにおける解析は下記の三つのPhaseを順に実行させた。

Phase 1: D1部分領域についての接触探索および接触力評価

Phase 2: D2部分領域についての変形評価

Phase 3: D1部分領域についての、Phase 1で評価した接触力を考慮した変形評価

ここで、Phase 2はPhase 1の終了を待たずに始められるが、Phase 3の開始にはPhase 1が終了しコントローラにて接触力の集計が完了することが前提となる。このため、D2部分領域の数がアナライザ数より多い場合は通常問題ないが、接触解析に関与するD1部分領域が非常に多くD2部分領域がアナライザ数より少ない場合にはPhase 3の開始時に待ち時間が生じることになる。

さらに、本研究の動的陽解法領域分割形並列接触解析コードにおいては、時間積分は下記の三つのモードを繰り返しながら進行する。

Fig. 6 Analyzer flowchart

Table 1 Parameters of mesh and partitioning

	CASE1	CASE2	CASE3
Number of total elements	373248	373248	373248
Number of total nodes	389017	391645	407413
Number of subdomains	512	512	512
Number of elements of one subdomain	729	729	729
Number of sliding interfaces	0	1	7
Number of subdomains including sliding interfaces	0	64	256
Number of total nodes on sliding interfaces	0	5402	37814

Fig. 7 Analysis domain

Fig. 8 Speedup by parallel processing

NEW: 各 S_i^n に対する $S_j^n (j \in J)$ を更新し, 解析を行う。

NORMAL: master 側探索範囲の更新は行わず, 直前の時間ステップの結果を基準とした local な探索のみ行う。

UPDATE: NORMAL と同様だが解析終了時に V_i の更新を行う。

NEW および UPDATE はそれぞれ 1 時間ステップ, NORMAL は数ステップ～数十ステップ行う。

アナライザのフローチャートを図 6 に示す。解析する部分領域における接触面の有無, 現在のモードおよび Phase によって処理が分岐する。

5. 性能評価

5.1 問題設定 き裂を有する固体中の弾性波伝ば解析を通して, 本研究の動的陽解法領域分割形並列接触解析コードの並列処理効率を調べた。具体的には, 図 7 に示す三つの場合 (Case 1 から Case 3) について解析し, 結果を比較した。いずれも底面四隅を固定し, 上面中央部から弾性波を入射した。要素数はいずれも六面体一次要素 373 248 個 ($72 \times 72 \times 72$), 領域分割は 512 領域 ($8 \times 8 \times 8$) に均等分割し一部分領域の要素数は 729 ($9 \times 9 \times 9$) である。Case 1 は接触面のない場合, Case 2 は一対の接触面 (接触面を有する領域数は 64, 一対の接触面の分割数は 64), Case 3 は七対の接触面 (接触面を有する領域数は 256, 一対の接触面の分割数は 64) を有する場合である。各接触面は初期状態では向かい合った接触面同士が距離零で接触している。材料は鋼を想定し, 時間刻みは $0.05 \mu s$ とした。Table 1 に各 Case における諸元を示す。

並列処理環境には Ethernet 100 BASE-TX で結合した 9 台のコンピュータ (CPU: Pentium II 350

MHz, Memory: 256 MB, OS: FreeBSD + pvm 3.3.11) からなる PC クラスタを用いた。コントローラに 1 台を割り当て, アナライザが 1 台, 2 台, 4 台, 8 台の場合について計算時間を測定した。

5.2 並列処理効率 図 8 に並列処理による加速率を示す。横軸は部分領域の解析に使用したアナライザ (プロセッサ) 数, 縦軸はアナライザ数 1 のときを基準とした加速率である。比較のために理想的な加速率 (アナライザ数と一致する) も示した。接触に関与する部分領域が多いほど加速率は低下しているものの, Case 3 (全領域の半数が接触に関与) の場合でも 8 CPU で 5 倍以上の加速率が得られていることがわかる。8 CPU 使用時の並列処理効率⁽¹²⁾ は, 各 Case についてそれぞれ 87.5%, 81.2%, 68.3% となった。使用した並列処理環境が比較的通信速度が遅く (約 3.4 MByte/s) レイテンシも大きいことが並列処理効率に影響していると考えられる。

6. 結 言

非定常・接触問題のための領域分割形並列有限要素解析コードを開発した。並列処理の基本アルゴリズムとしては領域分割法を採用しており, さまざまな並列処理環境においてスケラブルな処理性能を得ることができる。接触アルゴリズムに関しては master-slave algorithm に基づく領域分割形並列接触解析アルゴリズムを開発した。接触部分も自由に分割することが可能であるため, プリプロセスで行われる領域分割への負担が小さく, 解析時の負荷分散も容易である。本研究の領域分割形並列接触アルゴリズムを実装した動的陽解法領域分割形並列接触解析コードを開発し, き裂を含む固体中の弾性波伝ば解析問題を例として性能を検証した。その結果, 百万自由度規模の問題において通信速度の比較的遅い PC クラスタ上の解析でも 8 CPU で 5 倍以上の加速率が得られた。

なお, 本研究は日本学術振興会未来開拓学術研究推進事業「設計用大規模計算力学システムの開発」(JSPS-RFTF97P01104) の一環として行われたものである。

文 献

- (1) Lewis, T. G. and El-Rewini, H., *Introduction to parallel computing*, (1992), 1-56, Prentice-Hall.
- (2) Belytschko, T., *Comput. Struct.*, **42**(1992), 271-279.
- (3) Zhong, Z.-H., *Finite Element Procedures for Contact-Impact Problems*, (1993), 1-76, Oxford U. P.
- (4) Hallquist, J. O., ほか 2 名, *Comp. Meth. Appl. Mech. Eng.*, **51**(1985), 107-137.
- (5) Zhong, Z.-H. and Nilsson, L., *Comput. Struct.*, **33**

- (1989), 197-209.
- (6) Zhong, Z.-H. and Nilsson, L., *Comput. Struct.*, **34** (1990), 327-335.
- (7) Belytschko, T. and Neal, M. O., *Int. J. Num. Meth. Eng.*, **31**(1991), 547-572.
- (8) Belytschko, T. and Yeh, I. S., *Comp. Meth Appl. Mech. Eng.*, **105**(1993), 375-393.
- (9) Yagawa, G., ほか 2 名, *Comput. Struct.*, **38**(1991), 615-625.
- (10) Yagawa, G., ほか 3 名, *Comput. Struct.*, **47**(1993), 407-418.
- (11) 大石篤哉・ほか 3 名, 機論, **58**-552, A(1992), 1445-1452.
- (12) 大石篤哉・ほか 3 名, 機論, **62**-593, A(1996), 253-260.
- (13) Geist, A., ほか 5 名, *PVM : Parallel Virtual Machine*, (1994), 1-133, MIT Press.
- (14) Snir, M., ほか 4 名, *MPI : The Complete Reference*, (1996), 1-325, MIT Press.
- (15) Golub, G. H. and Van Loan, C. F., *Matrix Computations* (2nd ed.), (1989), 281-282, Johns Hopkins U. P.
- (16) 大石篤哉, 東京大学博士論文, (1996), 42-50.
- (17) Hoover, C. G., ほか 2 名, *UCRL-JC-130461, Lawrence Livermore National Laboratory*, (1998).
- (18) Benson, D. J. and Hallquist, J. O., *Comp. Meth. Appl. Mech. Eng.*, **78**(1990), 141-163.
- (19) Attaway, S. W., ほか 6 名, *Comput. Mech.*, **22**(1998), 143-159.
- (20) Malone, J. G. and Johnson, N. L., *Int. J. Num. Meth. Eng.*, **37**(1994), 591-603.
- (21) Elsner, B., ほか 4 名, *Proc. Parco95*, (1996), 157-164.
- (22) Plaskacz, E. J., *Proc. ICES'95*, (1995), 369-374.
- (23) Wang, S. P. and Nakamachi, E., *Int. J. Num. Meth. Eng.*, **40**(1997), 3665-3685.