

共有メモリー計算機における 3次元時間域動弾性高速境界積分方程式法の並列化について

On the parallelisation of a time domain fast boundary integral equation method for three dimensional elastodynamics for shared memory computers

大谷佳広*・西村直志**
Yoshihiro OTANI and Naoshi NISHIMURA

*学生会員 京都大学大学院工学研究科 社会基盤工学専攻 (〒606-8501 京都市左京区吉田本町)
**正会員 工博 京都大学学術情報メディアセンター教授 (〒606-8501 京都市左京区吉田本町)

This paper discusses parallelisation strategies of a time domain fast boundary integral equation method for three dimensional elastodynamics for a large shared memory parallel computer. The performance of the OpenMP parallelised code is examined. Also in the case of MPI-OpenMP hybrid parallelisation, a numerical solution of the scattering problem with more than one million DOFs is shown. It is concluded that the parallelisation with OpenMP is very effective, and that larger problem can be analysed with MPI-OpenMP hybrid parallelisation.

Key Words : Parallelisation, OpenMP, MPI, PWTB algorithm

1. はじめに

境界要素法で時間域の波動問題を解く場合、空間と時間の自由度をそれぞれ N_s 、 N_t とすると、従来法ならば計算量は $O(N_s^2 N_t^2)$ になる。従って従来法のままで、未知数を大きく取るとは不可能である。そこで、高速化に対する様々な試みがなされてきた。Erginらは3次元スカラー波動問題について、基本解の平面波展開と時空の階層化を利用した高速アルゴリズムの Plane Wave Time Domain(PWTD) アルゴリズムを提案した¹⁾。また、Luらは2次元スカラー波動問題についてのPWTD アルゴリズムの定式化を導いた²⁾。2次元スカラー波動問題の場合、計算量が $O(N_s N_t \log N_s \log N_t)$ になることが示されている。筆者らのグループではここ数年来、主に弾性問題の高速アルゴリズムについて研究を行ってきた。高橋らはPWTD アルゴリズムを2次元、3次元の弾性問題に拡張し空間と時間の自由度が各々 10^4 、 10^2 規模の問題を解析している^{3) 4)}。

近年、PC クラスターの普及に伴い、分散メモリー型並列計算機が身近になった。一方大型計算機の世界では大規模 SMP node を結合したスカラー計算機に移行しようとする流れが出てきている。例えば京都大学の学術情報メディアセンターにおいては、2004年3月に従来のベクトル並列計算機を、スカラー並列計算機に置き換えた。SMP型スカラー並列計算機では、各ノード内で大容量のメモリーが共有されるのが特徴である。また、スレッド並列の de facto standard である OpenMP⁵⁾ は並列化が非常に容易であるという長所を持つ。

そこで、本研究では高橋らが提案した3次元動弾性問題の高速アルゴリズムに OpenMP による並列化を施し、その有効性を検討する。また、MPI⁶⁾-OpenMP のハイブリッド並列を行うことで問題の大規模化を図り、

空間について自由度が100万を越える問題を解析した。

2. 定式化

2.1 3次元時間域動弾性問題とその境界積分方程式

本節と次節では本論文で取り扱う高速境界積分方程式法の概要について述べる。記述は並列化手法を示すための必要最小限にとどめるので、詳細は高橋他⁴⁾を参照して頂きたい。

三次元領域 $D \subset R^3$ において、以下の初期値境界値問題を解き、変位 $u_i(\mathbf{x}, t)$ を $\mathbf{x} = (x_1, x_2, x_3) \in D, t \in (0, \infty)$ で求めることを考える。

$$c_2^2 u_{i,jj}(\mathbf{x}, t) + (c_1^2 - c_2^2) u_{j,ij}(\mathbf{x}, t) + b_i(\mathbf{x}, t) = \ddot{u}_i(\mathbf{x}, t) \quad (1)$$

$$\text{初期条件 } \mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}) \quad \text{in } D \quad (2)$$

$$\dot{\mathbf{u}}(\mathbf{x}, 0) = \mathbf{v}_0(\mathbf{x}) \quad \text{in } D \quad (3)$$

$$\text{境界条件 } \mathbf{u}(\mathbf{x}, t) = \bar{\mathbf{u}}(\mathbf{x}, t) \quad \text{on } S_1 \quad (4)$$

$$\mathbf{t}(\mathbf{x}, t) = \bar{\mathbf{t}}(\mathbf{x}, t) \quad \text{on } S_2 \quad (5)$$

ここに、 $\bar{(\cdot)}$ は境界で与えられた関数である。また、 $S_1 \subset S, S_2 \subset S, S_1 + S_2 = S$ である。 x_i に関する空間微分は $(\cdot)_{,i}$ と表記した。 \mathbf{u}_0 と \mathbf{v}_0 は各々初期変位、初期速度を表すが今回は零とする。 \mathbf{t} は表面力であり、以下のように与えられる。

$$t_i(\mathbf{x}, t) = C_{ijkl} n_j(\mathbf{x}) u_{k,i}(\mathbf{x}, t) \quad (6)$$

ここに、 \mathbf{n} は境界 D の補領域方向を向く単位法線ベクトルである。 C_{ijkl} は弾性定数テンソルであり、Lamé 定数 λ, μ を用いて以下のように書ける。

$$C_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu (\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk}) \quad (7)$$

† Dedicated to the memory of Prof. Michihiro KITAHARA

c_1 、 c_2 はそれぞれ P 波速度、S 波速度であり、

$$c_1 = \sqrt{\frac{\lambda + 2\mu}{\rho}} \quad (8)$$

$$c_2 = \sqrt{\frac{\mu}{\rho}} \quad (9)$$

の関係が成り立つ。

簡単のため初期条件、物体力を零とすると領域 D の境界 S で以下の積分方程式が成り立つ。

$$\frac{1}{2}u_i + \int_S T_{ij}(\mathbf{x}, \mathbf{y}, t) * u_j(\mathbf{y}, t) dS_y = \int_S \Gamma_{ij}(\mathbf{x} - \mathbf{y}, t) * t_j(\mathbf{y}, t) dS_y \quad \mathbf{x} \in S \quad (10)$$

ここに、 \int は Cauchy の主値積分を表し、 $*$ は時間に関する畳み込みの意である。 Γ ならびに t はそれぞれ動弾性学の基本解及び二重層核であり、各々

$$\Gamma_{ij}(\mathbf{x}, t) = \frac{1}{4\pi\mu} \left[\frac{\delta(t - |\mathbf{x}|/c_2)}{|\mathbf{x}|} \delta_{ij} - c_2^2 \partial_i \partial_j \left(\frac{(t - |\mathbf{x}|/c_2)_+}{|\mathbf{x}|} - \frac{(t - |\mathbf{x}|/c_1)_+}{|\mathbf{x}|} \right) \right] \quad (11)$$

$$T_{ij}(\mathbf{x}, \mathbf{y}, t) = C_{jlmn} n_l(\mathbf{y}) \frac{\partial}{\partial y_n} \Gamma_{im}(\mathbf{x} - \mathbf{y}, t) \quad (12)$$

の様に表示される。ここに δ_{ij} は Kronecker のデルタ、 δ は Dirac のデルタである。また、 $f_+ = (|f| + f)/2$ である。

2.2 従来法による境界積分方程式の離散化

式 (10) を次のように離散化する。境界 S を N_s 個の区分直線 $S_I (I = 1, 2, \dots, N_s)$ で分割し各 S_I の中央に境界選点 \mathbf{x}_I を設ける。時間については N_t 個の区間 $(t_{\alpha-1}, t_\alpha)$ で分割する。各区間の長さは Δt で一定とし、各区間の時間選点は区間終端に設ける。選点における境界密度、すなわち変位 $\mathbf{u}(\mathbf{x}_I, t_\alpha)$ 及び表面力 $\mathbf{t}(\mathbf{x}_I, t_\alpha)$ をそれぞれ \mathbf{u}_I^α 及び \mathbf{t}_I^α と記す。

境界密度の基底として区分一定のものを、時間基底として区分線形のものを用いる。時間基底は次のように表される。

$$\Psi_\alpha(t) = \begin{cases} 0 & t < t_{\alpha-1}, t_{\alpha+1} \leq t \\ \frac{t - t_{\alpha-1}}{\Delta t} & t_{\alpha-1} \leq t < t_\alpha \\ \frac{t_{\alpha+1} - t}{\Delta t} & t_\alpha \leq t < t_{\alpha+1} \end{cases} \quad (13)$$

これより、 $\mathbf{u}(\mathbf{x}, t)$ 、 $\mathbf{t}(\mathbf{x}, t)$ は時間に関して

$$\mathbf{u}(\mathbf{x}, t) = \sum_{\alpha=1}^{N_t} \Psi_\alpha(t) \mathbf{u}(\mathbf{x}, t_\alpha) \quad (14)$$

$$\mathbf{t}(\mathbf{x}, t) = \sum_{\alpha=1}^{N_t} \Psi_\alpha(t) \mathbf{t}(\mathbf{x}, t_\alpha) \quad (15)$$

と内挿される。ここで

$$\mathbf{W}_{IJ}^{\alpha+\beta-1} = \frac{1}{2} \delta_{IJ} \delta_{\alpha\beta} \mathbf{I} + \int_{t_{\beta-1}}^{\min(t_{\beta+1}, t_\alpha)} \int_{S_J} \mathbf{T}(\mathbf{x}_I, \mathbf{y}, t_\alpha - s) \Psi_\beta(s) dS_y ds \quad (16)$$

$$\mathbf{U}_{IJ}^{\alpha+\beta-1} = \int_{t_{\beta-1}}^{\min(t_{\beta+1}, t_\alpha)} \int_{S_J} \mathbf{\Gamma}(\mathbf{x}_I - \mathbf{y}, t_\alpha - s) \Psi_\beta(s) dS_y ds \quad (17)$$

として影響係数を定義する。すると式 (10) は次のように離散化される。

$$\sum_{J=1}^{N_s} \mathbf{A}_{IJ}^{(1)} \mathbf{p}_J^\alpha = \sum_{J=1}^{N_s} \mathbf{B}_{IJ}^{(1)} \mathbf{q}_J^\alpha - \sum_{\beta=1}^{\alpha-1} \sum_{J=1}^{N_s} \left(\mathbf{W}_{IJ}^{(\alpha-\beta+1)} \mathbf{u}_J^\beta - \mathbf{U}_{IJ}^{(\alpha-\beta+1)} \mathbf{t}_J^\beta \right) \quad (18)$$

ここに、現在時刻 t_α の境界要素 S_J における境界密度の未知成分を \mathbf{p}_J^α 、境界条件によって与えられる成分を \mathbf{q}_J^α と表し、その係数 ($\mathbf{W}_{IJ}^{(1)}$ 、 $\mathbf{U}_{IJ}^{(1)}$ のいずれか) をそれぞれ $\mathbf{A}_{IJ}^{(1)}$ 、 $\mathbf{B}_{IJ}^{(1)}$ と書いた。式 (18) は $3N_s$ 元の連立一次方程式である。 $\mathbf{A}_{IJ}^{(1)}$ は一般に疎行列であるため、この逆行列はおおよそ $O(N_s)$ で求まる。しかし、式 (18) の第 2 項の $\mathbf{U}_{IJ}^{(\gamma)}$ 、 $\mathbf{W}_{IJ}^{(\gamma)}$ は時間差 γ の増加とともに密になる。また、行列ベクトル積の回数は時間に比例する。従って時刻ステップ N_t の解 $\mathbf{p}_I^{N_t}$ を求めるのに従来法が必要とする計算量は $O(N_s^2 N_t^2)$ である。

解法を高速化するためには式 (18) の右辺の行列・ベクトル積を高速化できれば良い。そこで、基本解の平面波展開を用いた高速解法によりこの演算を高速化する。

2.3 基本解の平面波展開

基本解 $\mathbf{\Gamma}$ を次の様に平面波展開することができる。

$$\begin{aligned} \Gamma_{ij}(\mathbf{x}, t) - \Gamma'_{ij}(\mathbf{x}, t) &= -\frac{\partial t}{8\pi^2} \int_{S_k} \left[\frac{k_i k_j}{\rho c_1^3} \delta(t - \mathbf{x} \cdot \mathbf{k}/c_1) \right. \\ &\quad \left. + \frac{k_p k_q e_{pik} e_{qjk}}{\rho c_2^3} \delta(t - \mathbf{x} \cdot \mathbf{k}/c_2) \right] dS_k \quad (19) \end{aligned}$$

ここに、 S_k は単位球面、 \mathbf{k} は S_k 上の外向き単位法線ベクトルである。また $\mathbf{\Gamma}' = \mathbf{\Gamma}(\mathbf{x}, -t)$ は反因果律を満足する動弾性学の基本解である。式 (19) が三次元動弾性学の基本解 $\mathbf{\Gamma}$ の平面波展開である。

2.4 平面波展開による層ポテンシャルの評価

式 (19) を用いて式 (10) の層ポテンシャルの高速算法を構成する。

いま、ともに半径が R である二つの球形領域 S_s 、 S_o を考え、各々の中心を \mathbf{s} 、 \mathbf{o} とする。中心間距離 $|\mathbf{o} - \mathbf{s}|$

を R_c とし $R_c > 2R$ を満たすものとする。また、 S_s に含まれる境界 S の部分を S_0 とする。そこで、式 (19) を用いて選点 (\mathbf{x}, t) ($\mathbf{x} \in S_0, t \in (0, \infty)$) において $S_0 \times (0, t]$ に分布した密度 \mathbf{t} 、 \mathbf{u} による層ポテンシャルを評価することを目標とする。

密度 \mathbf{u} 及び \mathbf{t} を各々有限の時間区間帯 $(T_1^z, T_2^z]$ のみを台に持つ関数 \mathbf{u}^z ならびに \mathbf{t}^z に区分けする。P 波速度を c_1 、S 波速度を c_2 とする。すると、

$$R_c - 2R \geq c_1(T_2^z - T_1^z) \quad (20)$$

が成り立てば選点 (\mathbf{x}, t) ($\mathbf{x} \in S_0, t > T_2^z$) に対して $S_0 \times (T_1^z, T_2^z]$ 上の既知の密度 \mathbf{t}^z ならびに \mathbf{u}^z によるポテンシャルは次のように計算できる⁴⁾。

$$\begin{aligned} & \int_{S_0} (T_{ij}(\mathbf{x}, \mathbf{y}, t) * u_j^z(\mathbf{y}, t) - \Gamma_{ij}(\mathbf{x} - \mathbf{y}, t) * t_j^z(\mathbf{y}, t)) dS_y \\ &= -\frac{\partial_t}{8\pi^2} \int_{S_k} [k_i \delta(t - (\mathbf{x} - \mathbf{s}) \cdot \mathbf{k} / c_1) * O^z(\mathbf{s}, t, \mathbf{k}) \\ & \quad + e_{pik} k_p \delta(t - (\mathbf{x} - \mathbf{s}) \cdot \mathbf{k} / c_2) * O_k^z(\mathbf{s}, t, \mathbf{k})] dS_k \end{aligned} \quad (21)$$

ここに、 O^z 及び O_k^z ($k = 1, 2, 3$) は outgoing ray であり、

$$\begin{aligned} O^z(\mathbf{s}, t, \mathbf{k}) &= \int_{S_0} \left(\frac{C_{jlnm} n_l k_m k_n}{\rho c_1^4} \dot{u}_j^z(\mathbf{y}, t - (\mathbf{s} - \mathbf{y}) \cdot \mathbf{k} / c_1) \right. \\ & \quad \left. - \frac{k_j}{\rho c_1^3} t_j^z(\mathbf{y}, t - (\mathbf{s} - \mathbf{y}) \cdot \mathbf{k} / c_1) \right) dS_y \end{aligned} \quad (22)$$

$$\begin{aligned} O_k^z(\mathbf{s}, t, \mathbf{k}) &= \int_{S_0} \left(\frac{C_{jlnm} n_l e_{qmk} k_q k_n}{\rho c_2^4} \dot{u}_j^z(\mathbf{y}, t - (\mathbf{s} - \mathbf{y}) \cdot \mathbf{k} / c_2) \right. \\ & \quad \left. - \frac{e_{qjk} k_q}{\rho c_2^3} t_j^z(\mathbf{y}, t - (\mathbf{s} - \mathbf{y}) \cdot \mathbf{k} / c_2) \right) dS_y \end{aligned} \quad (23)$$

と定義される。また、式 (21) は次のようにも計算される。

$$\begin{aligned} & \int_{S_0} (T_{ij}(\mathbf{x}, \mathbf{y}, t) * u_j^z(\mathbf{y}, t) - \Gamma_{ij}(\mathbf{x} - \mathbf{y}, t) * t_j^z(\mathbf{y}, t)) dS_y \\ &= -\frac{1}{8\pi^2} \int_{S_k} [k_i \delta(t - (\mathbf{x} - \mathbf{o}) \cdot \mathbf{k} / c_1) * I^z(\mathbf{o}, t, \mathbf{k}) \\ & \quad + e_{pik} k_p \delta(t - (\mathbf{x} - \mathbf{o}) \cdot \mathbf{k} / c_2) * I_k^z(\mathbf{o}, t, \mathbf{k})] dS_k \end{aligned} \quad (24)$$

ここに I^z 及び I_k^z ($k = 1, 2, 3$) は incoming ray であり、

$$I^z(\mathbf{o}, t, \mathbf{k}) = \partial_t \delta(t - (\mathbf{o} - \mathbf{s}) \cdot \mathbf{k} / c_1) * O^z(\mathbf{s}, t, \mathbf{k}) \quad (25)$$

$$I_k^z(\mathbf{o}, t, \mathbf{k}) = \partial_t \delta(t - (\mathbf{o} - \mathbf{s}) \cdot \mathbf{k} / c_2) * O_k^z(\mathbf{s}, t, \mathbf{k}) \quad (26)$$

と計算される。outgoing ray、ならびに incoming ray は次式により原点を移動できる。

$$O_{(k)}^z(\mathbf{s}, t, \mathbf{k}) = \delta(t - (\mathbf{s} - \mathbf{s}') \cdot \mathbf{k} / c_{1(2)}) * O_{(k)}^z(\mathbf{s}', t, \mathbf{k}) \quad (27)$$

$$I_{(k)}^z(\mathbf{s}, t, \mathbf{k}) = \delta(t - (\mathbf{s} - \mathbf{s}') \cdot \mathbf{k} / c_{1(2)}) * I_{(k)}^z(\mathbf{s}', t, \mathbf{k}) \quad (28)$$

3. 高速解法のアルゴリズム

従来法による積分と 2.4 で導いた平面波展開の双方を利用し、式 (10) の右辺の層ポテンシャルを計算する手法を構成する。すなわち、ある境界選点について遠方の要素からの寄与は平面波展開を用い近傍の要素からの影響は従来法で行う。

3.1 セルと時間区間

準備段階として、空間を階層構造を持つセルに分割する。まず領域 D を含むような立方体を作りこれを level 0 のセルとする。次にこれを 8 つの立方体に分割し level 1 のセルを作る。さらに level 1 のセルを 8 分割し level 2 のセルを作る。この分割を同様に繰り返すが、中に含まれる要素が所定の数を下回っているセルについては分割を行わない。これを、分割できるセルがなくなるまで繰り返す。最下層のレベルを l_{max} という。子を持たないセルをリーフと言う。

近傍の定義は次に様にして行う。ある level l のセル C と C' の中心座標をそれぞれ C_i 、 C'_i とする。level l のセルの一辺の長さを $L^{(l)}$ とすると

$$|C_i - C'_i| < (\beta + 1)L^{(l)} \quad i = 1, 2, 3 \quad (29)$$

が成立する時 C と C' は近いと言い、上式が成立しない時 C と C' は遠いと言う。今回は $\beta = 1$ とした。すなわちセルとセルの間に一つ以上セルが挟まっていれば遠いと判定する。

遠方からの寄与は平面波展開を用いるがその際、式 (20) が満たされる必要がある。そこで時間区間の概念を導入する。時間軸を $M\Delta t$ 毎に分割し、区間 $((z-1)M\Delta t, zM\Delta t]$ ($z = 1, 2, \dots$) を第 z 時間区間と呼ぶ。この時間区間により境界密度 \mathbf{u} 、 \mathbf{t} をそれぞれ \mathbf{u}^z 、 \mathbf{t}^z に区分けする。ただし、離散化された境界密度を台が $2p_t$ である補間関数によって補間するので第 z 時間区間の境界密度の台 $(T_1^z, t_2^z]$ は

$$T_1^z = t((z-1)M + 1 - p_t)\Delta t \quad (30)$$

$$T_2^z = (zM + p_t)\Delta t \quad (31)$$

によって与えられる。各レベルにおいて遠方のセルからの影響を平面波展開によって評価できるためには、式 (20) を考慮すると M を各レベル毎に次のように取れば良い。

$$M^{(l_{max})} = \frac{\beta L^{(l_{max})}}{c_1 \Delta t} - 2p_t + 1 \quad (32)$$

$$M^{(l)} = 2M^{(l+1)} \quad l = 2, 3, \dots, l_{max} - 1 \quad (33)$$

この $M^{(l)}$ によってレベル毎に時間区間 $z^{(l)}$ が定義され、式 (30)、(31) によって $T_1^{z^{(l)}}$ 、 $T_2^{z^{(l)}}$ が与えられる。

3.2 高速解法の計算手順

以下に平面波展開を用いた高速解法の計算手順を示す。

1. $S \times (0, t_{\alpha-1}]$ 上の密度による層ポテンシャルの評価
 現在時刻を $t_{\alpha} = \alpha \Delta t$ とする。その時、 $S \times (0, t_{\alpha-1}]$ 上の既知密度による層ポテンシャルを近傍からの寄与と遠方からの寄与に分けて評価する。
 - 近傍からの寄与
 近傍からの寄与は従来法による直接計算によって評価する。
 - 遠方からの寄与
 現時刻 t_{α} が level l の第 $z_{\alpha}^{(l)}$ 時間区間に属しているとする。すると、過去の時間区間 $z^{(l)} = 1, 2, \dots, z^{(l)\alpha} - 1$ の境界密度 $\mathbf{u}^{z^{(l)}}, \mathbf{t}^{z^{(l)}}$ による outgoing ray $O_p^{z^{(l)}}$ 、incoming ray $I_p^{z^{(l)}}$ は既知であり、これらによる寄与は後述の 3(b) において式 (24) によって既に評価されている。従って記憶された値を呼び出すだけである。なお、現時刻が属する時間区間の境界密度からの寄与は現時刻には到達しないので考慮する必要はない。
 2. 現時刻の未知境界密度の求解
 式 (18) の右辺は 1 によって得られた。そこで式 (18) を反復解法によって解き現時刻での未知成分を求める。
 3. outgoing ray 及び incoming ray の計算
 現時間区間 $z_{\alpha}^{(l)}$ の境界密度による outgoing ray 及び incoming ray を計算する。
 - (a) upward pass
 最下層の level l_{max} から上に向かって level 2 に向かって現時間区間 $z_{\alpha}^{(l)}$ の各セル中心に関する outgoing ray を計算する。最上レベルが 2 であるのは、level 1 以上のセルは全て近接しており平面波展開が使えないからである。ただし、現時刻ステップ数 α が $M^{(l)}$ の整数倍である level l についてのみ実行する。リーフについては式 (22)、(23) によって outgoing ray を求め、子がいる場合は式 (27) によって求める。
 - (b) downward pass
 level 2 から下向きに各レベルで現時間区間 $z_{\alpha}^{(l)}$ の各セル中心に関する incoming ray を計算する。ただし、現時刻ステップ数 α が $M^{(l)}$ の整数倍である level l についてのみ実行する。遠方のセルの outgoing ray から式 (25) ならびに式 (26) によって incoming ray を計算する。子がいる場合は式 (28) を用いて子の中心まわりの incoming ray を計算し、子がない場合は式 (24) を用いてそのセルの各選点 $(\mathbf{x}, t_{\alpha'})$ ($\alpha' = \alpha + 1, \dots, N_t$) でポテンシャルを評価する。
- なお、incoming ray、outgoing ray の第三引数については単位球面上で定義されるが、数値計算では level l においては $(2K^{(l)} + 1)(K^{(l)} + 1)$ 個の離散点においてサンプリングする。 $K^{(l)}$ は $K^{(l)} = 2K^{(l+1)}$ を満たすものとする。
4. 更新
 時間ステップを一つ進める

4. 並列化

本節では、以上で述べた高速境界積分方程式法を並列化する。

4.1 OpenMP による並列化

OpenMP においては並列化の効果が上がる部分のみを並列化することが容易である。本論文では計算量の多い

- 直接計算による係数行列の評価
- upward pass
- downward pass

の部分と並列化した。連立一次方程式の反復解法には GMRES を用いたが、これに関する部分は並列化を行っていない。この理由は、GMRES で最も計算コストがかかる部分である係数行列・ベクトル積演算が、係数行列、つまり式 (18) の $\mathbf{A}_{IJ}^{(1)}$ が疎行列であるために非常に小さなコストで実行できると判断されるためである。

(1) 直接計算部分

直接計算による影響係数の評価に関わる部分の並列化は以下のように行った。Fortran 風の擬コードで記す。

```
do level=最下層のレベル,2,-1
!$OMP PARALLEL DO
  do 当該レベルのセル
    if(当該セルがリーフ) then
      当該セルの近傍から影響係数を評価
    endif
  enddo
!$OMP END PARALLEL DO
enddo
```

(2) upward pass

upward pass の並列化は以下のように行った。

```
do level=最下層のレベル,2,-1
  if(現時刻ステップ数が M の整数倍なら)
!$OMP PARALLEL DO
    do 当該レベルのセル
      if(当該セルがリーフ) then
        リーフ内各選点の境界密度から
          outgoing ray を生成
      else
        子から outgoing ray を移動
      endif
    enddo
!$OMP PARALLEL DO
  endif
enddo
```

なお、子から親へ outgoing ray を渡す際には、子の側でループを回す方法と親の側でループを回す方法の二通りが考えられるが、今回は上のように後者を採用している。それは、子の側でループを回した場合には同一の親を持つセルが異なるスレッドに割り当てられるため、スレッド間で親の outgoing ray への書き込みが競合してしまうからである。

(3) downward pass

downward pass の並列化は以下のように行った。

```
do level=2, 最下層のレベル
  if(現時刻ステップ数が M の整数倍なら)
!$OMP PARALLEL DO
  do 当該レベルのセル

    親の近傍で自分の近傍でないセルの
    outgoing ray から incoming ray を生成

    if(当該セルがリーフ)then
      incoming ray からセル内の各選点で
      ポテンシャルを計算
    else
      子に incoming ray を移動
    endif
  enddo
!$OMP PARALLEL DO
  endif
enddo
```

outgoing ray から incoming ray を生成する部分は upward pass に置く方法もあるが、異なるスレッド間の書き込みの競合を防ぐと言う観点から downward pass においた。

4.2 MPI と OpenMP のハイブリッド並列

MPIによるプロセス並列化は、領域 D を分割して各 CPU に割り当てる方法を取った。領域の分割は level 2 において行った。level 2 のセルは 64 個あるのでプロセス数を m とすると一つのプロセスあたり $64/m$ 個のセルのカバーする領域が割り当てられる。それぞれのプロセスは、incoming ray や outgoing ray を自分の担当領域の分のみ計算し保持する。直接計算される近傍からの影響係数についても、自分の担当領域内の選点においてのみ計算、保持する。反復解法に GMRES を用いたが、これに関しては並列化していない。

次に OpenMP によるスレッド並列に MPI によるプロセス並列をどのように組み合わせてハイブリッド並列化するかについて、小節 4.1 で記した擬コードの変更点を述べることによって説明する。

前述のとおり各プロセスは自分の担当する領域の選点に関する影響係数、ならびに担当する領域のセルに関する outgoing ray、incoming ray のみ計算して記憶する。従って、小節 4.1 の擬コードにおけるセルのループはすべて、担当する領域内のセルだけで回すことに変更する。このループはプロセス内では OpenMP によってスレッド並列化されている。

downward pass については outgoing ray から incoming ray を生成する部分で、相手のセルが自分のプロセスの担当する領域に存在しない場合があり、プロセス間の通信が必要になる。ただしセルのループの中で通信をしていたのでは通信回数が多くなりすぎる。従って通信回数を減らすという観点から、担当する領域内の incoming ray を生成するのに通信が必要になるケースをあらかじめリストアップしておき、downward pass

の最初に必要なプロセス間通信を一括して行うことにする。

以上の手順により式 (18) の右辺が各プロセスに分割された形で求まる。最後に通信によってそれらを集め、式 (18) の右辺を完成する。

影響係数は各プロセスの担当する領域内の選点に関してのみ計算されているので、式 (18) の右辺の係数行列は横割されて各プロセスに保持されている。従って、GMRES で係数行列・ベクトル積演算が必要になる場合はその都度通信する必要がある。

5. 数値解析

OpenMP による並列化の有効性を調べる。また、MPI-OpenMP のハイブリッド並列を行うことにより問題の大型化を図る。

5.1 計算環境

数値計算は京都大学学術情報メディアセンターの HPC2500 にて行った。HPC2500 は 11 台の計算ノードからなり、ノード間は高速光インターコネクトを用いて高速接続されている。ノード内では 512GB のメモリーが共有される。ノード当たりの CPU 台数は 128、CPU は SPARC64V(1.56GHz) である。

5.2 数値計算例

OpenMP を用いたスレッド並列の並列化効率を調べるために直方体の内部問題と球孔群の外部問題を解析した。直方体のケースに関しては解析解との比較により精度を検証した。球孔群のケースについては MPI を用いたプロセス並列の計算時間や MPI-OpenMP ハイブリッド並列の計算時間も調べた。問題の大型化の可能性を検討するため球孔群の外部問題について、球孔の数を増やしてさらに 2 ケース解析した。

(1) 直方体の内部問題

点 $(0, 0, 0)$ と $(1.40, 0.68, 0.80)$ を対角線の両端にもつ直方体の内部問題を扱う。

等辺が 0.02 の直角二等辺三角形で要素分割した。境界要素数は 26160(78480 DOF) である。図-1 に解析に用いた直方体を示す。

適当な無次元化を行い、直方体の密度 $\rho = 1$ 、P 波速度 $c_1 = 1$ 、S 波速度 $c_2 = 1/\sqrt{2}$ とした。境界条件は次式で与えられる変位場に対応する応力を直方体の全ての表面に与えた。

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{d} \left[1 - \text{Cos} \frac{2\pi}{\Lambda} \left(t - \frac{\mathbf{d} \cdot \mathbf{x}}{c_1} \right) \right] \quad (34)$$

ここに、 \mathbf{d} は入射波の進行方向を示し $\mathbf{d} = (0, 0, 1)$ である。また、 $\Lambda = 1.0$ とした。Cos の定義は次式による。

$$\text{Cos} x = \begin{cases} \cos x & 0 \leq x \leq 2\pi \\ 1 & x < 0, x > 2\pi \end{cases} \quad (35)$$

時刻ステップの間隔 $\Delta t = 0.01$ 、時刻ステップ数 $N_t = 200$ である。リーフの最大要素数は 150 とした。また、level 2 における incoming ray ならびに outgoing ray のサンプリング点数は $K^{(2)} = 56$ により与えた。

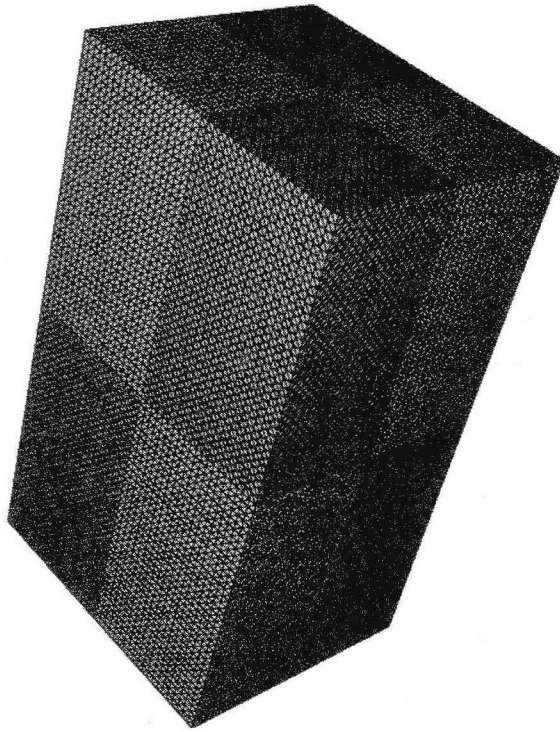


図-1 直方体の要素分割

図-2 に OpenMP におけるスレッド並列数と計算時間の関係を、図-3 にスレッド並列数と speed up との関係を示す。なお、speed up は (スレッド並列数が 1 の時の計算時間)/(計算時間) で定義される。理想的には speed up は並列数になる。グラフ中に理想値を点線で示した。

式 (34) で与えられる解析解と数値解を時刻ステップ毎に比較したところ誤差は最大値を示す要素でも、3.28% であり、精度は十分であると考えられる。

スレッド数が 64 の時計算時間は最短であり、スレッド数が 128 になると計算時間は逆に増加している。このことについて、アルゴリズム面では以下のような理由が考えられる。level 2 は、incoming ray 及び outgoing ray のサンプリング点数が最も多く、時間区間の長さも最も長い。従って他の level に比べ計算コストが高い。しかしながら level 2 のセル数は最大でも $8^2 = 64$ であるので、スレッド数をそれ以上にしても最もコストが高い level 2 では全スレッドが有効に使われない。従ってスレッド数 64 を境に急激に並列化効率が悪くなると考えられる。ハード面での理由としては、スレッド数が多くなりすぎたことによりメモリアクセスの競合が顕在化したことが考えられる。

(2) 球孔群の問題

次に、無限弾性体中の球孔群による散乱問題を扱う。

球孔は x_1, x_2, x_3 方向にそれぞれ 4 個ずつ並んでいる。隣接する球孔の中心間距離は $2a = 0.15$ 、球孔の半径は $0.9a = 0.0675$ である。各球孔は一辺が約 0.02 の三角形要素で分割した。一つの球孔当たりの要素数は 320、全体では 20480 要素 (61440 DOF) である。図-4

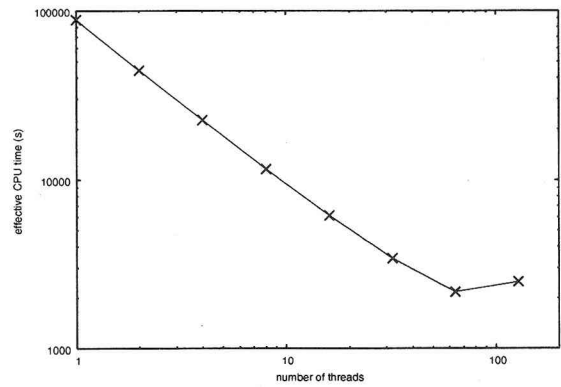


図-2 OpenMP による計算時間 (直方体の問題)

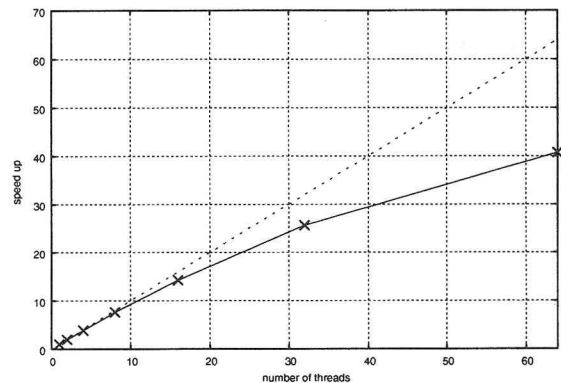


図-3 OpenMP による speed up (直方体の問題)

に球孔群を示す。

用いた定数は弾性体の密度 $\rho = 1$ 、P 波速度 $c_1 = 1$ 、S 波速度 $c_2 = 1/\sqrt{2}$ で、直方体の場合と同じである。

時刻ステップの間隔 $\Delta t = 0.01$ 、時刻ステップ数 $N_t = 200$ である。リーフの最大要素数は 100 とした。また、 $K^{(2)} = 24$ とした。

入射波 \mathbf{u}^I は次式により与えた。

$$\mathbf{u}^I = \mathbf{d} \left[1 - \text{Cos} \frac{2\pi}{\Lambda} \left(t - \frac{\mathbf{d} \cdot \mathbf{x}}{c_1} \right) \right] \quad (36)$$

入射方向 $\mathbf{d} = (0, 0, 1)$ 、 $\Lambda = 4a = 0.3$ とした。

図-5 に OpenMP によるスレッド並列計算におけるスレッド並列数と計算時間の関係を示す。また、参考として MPI によるプロセス並列のみを施した場合のプロセス並列数と計算時間の関係もあわせてプロットする。図-6 には OpenMP によるスレッド並列計算におけるスレッド並列数と speed up との関係を示す。表-1 には MPI-OpenMP ハイブリッド並列時の計算時間を示す。

スレッド数が 128 になったとき計算時間が増加しているのは前項 (1) で述べた理由によるものと考えられる。

図-5 から、MPI による並列化時よりも OpenMP による並列化時の方が高速に計算できていることが分かる。また、表-1 より、 1×64 並列時が最も速く、続いて 8×8 並列が速いと言う結果になった。当然、解析時間は並列化、特に MPI による並列化の能率に左右さ

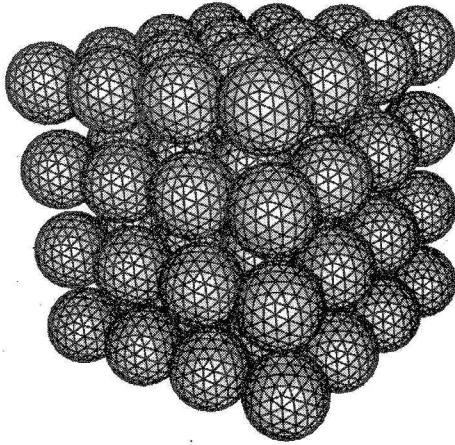


図-4 球孔群の要素分割

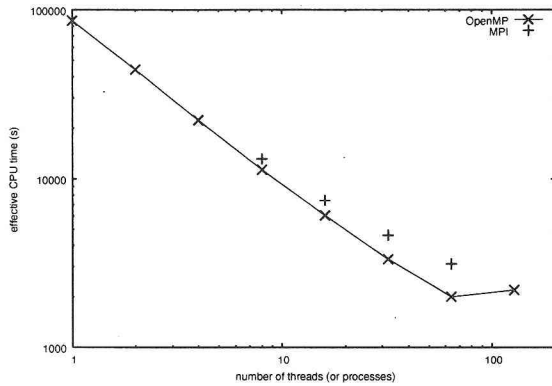


図-5 OpenMP による計算時間（球孔群の問題）

れる部分が大きく、今回の結果を必ずしも一般化できるものではない。実際、西村ら⁷⁾は直接法のみ境界要素法のハイブリッド並列化において、MPIの並列化の割合を高くする程計算が速くなることを示している。とはいえ、OpenMPによるスレッド並列化はMPIによるプロセス並列化に比べ比較的少ない開発労力で同程度の並列化効率を上げることができると言う点では、実用性が高いと考えられる。

(3) 大型化—その1

次に(2)と同様の問題設定で球孔の数を増やして解析を行った。球孔を x_1, x_2, x_3 方向にそれぞれ7個ずつ並べた。境界要素数は109760(329280 DOF)である。

弾性体の密度 $\rho = 1$ 、P波速度 $c_1 = 1$ 、S波速度 $c_2 = 1/\sqrt{3}$ とした。

時刻ステップの間隔 $\Delta t = 0.01$ 、時刻ステップ数 $N_t = 200$ である。リーフの最大要素数は100とした。また、 $K^{(2)} = 52$ とした。

入射波は式(36)においてCosを下で定義される関数COSで置き換え、連続的なパルス入射としたもの

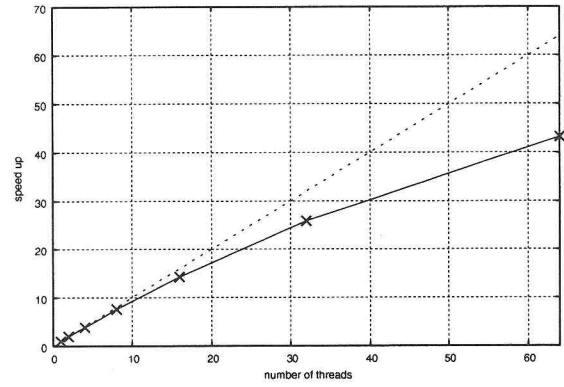


図-6 OpenMPによるspeed up（球孔群の問題）

表-1 MPI-OpenMP ハイブリッド並列による計算時間（球孔群の問題）

プロセス数	スレッド数	実効 CPU 時間 (s)
1	64	2053
2	32	2379
4	16	2356
8	8	2301
16	4	2603
32	2	2956
64	1	2947

である。

$$\text{COS}x = \begin{cases} \cos x & 0 \leq x \\ 1 & x < 0 \end{cases} \quad (37)$$

OpenMPによりスレッド並列化を行い64CPUで計算した所、要した実効CPU時間の最大値は12時間26分であった。球孔表面の変位を図-7から図-16に示す。

本問題は解析解が明らかでなく、またこの規模の問題を解析した先行研究がないために解析の精度は不明である。しかしながら、入射波の進行方向に対して直角方向には変位の対称性がなければならないが、数値解では対称性は数値誤差を除いて満たされており、入射波も正しい速度で伝播していることがわかる。その他、特に物理的に異常な点も見当たらないので、数値解の最低限の妥当性の確認はとれたものと考えられる。

(4) 大型化—その2

最後に球孔を x_1, x_2 方向にそれぞれ12個ずつ配し、 x_3 方向には8列並べ解析を行った。境界要素数は368640(1105290 DOF)である。

P波速度 $c_1 = 1$ 、S波速度 $c_2 = 1/\sqrt{2}$ とした。また、 $K^{(2)} = 80$ とした。その他の解析条件は(3)に同じである。

プロセス並列8、スレッド並列32、合計256CPUのハイブリッド並列で計算を行った。この規模の並列計算はスレッド並列では実行できずハイブリッド並列計算を必要とする。

球孔表面の変位を図-17から図-26に示す。数値解析結果は前項(3)で挙げた妥当性の確認事項を満たし

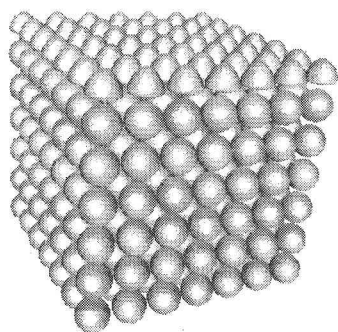


图-7 $t = 20\Delta t$

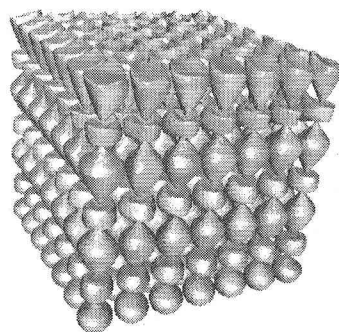


图-11 $t = 100\Delta t$

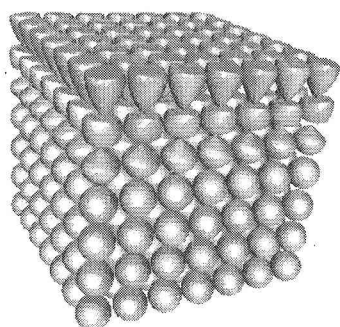


图-8 $t = 40\Delta t$

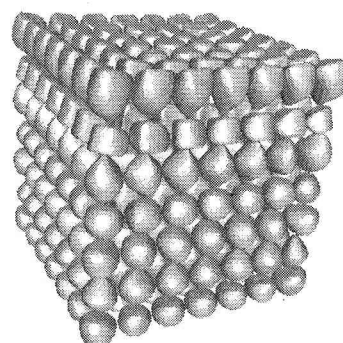


图-12 $t = 120\Delta t$

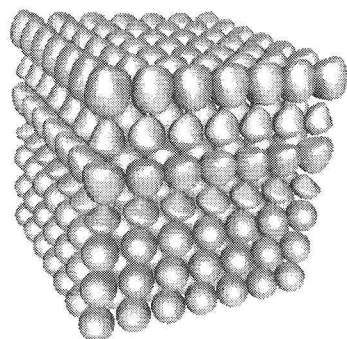


图-9 $t = 60\Delta t$

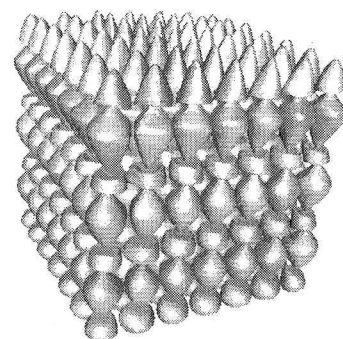


图-13 $t = 140\Delta t$

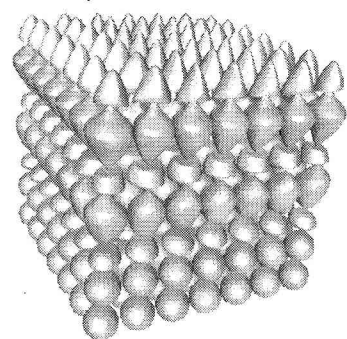


图-10 $t = 80\Delta t$

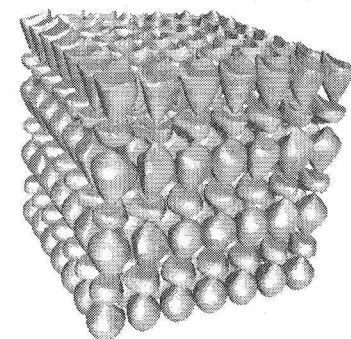


图-14 $t = 160\Delta t$

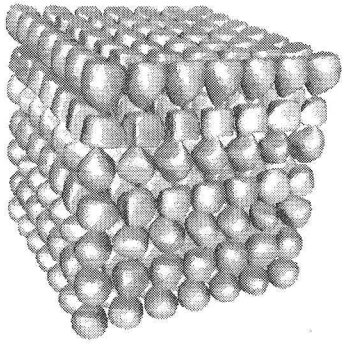


図-15 $t = 180\Delta t$

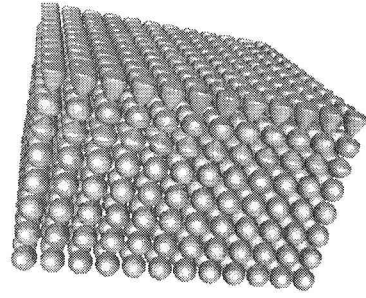


図-18 $t = 40\Delta t$

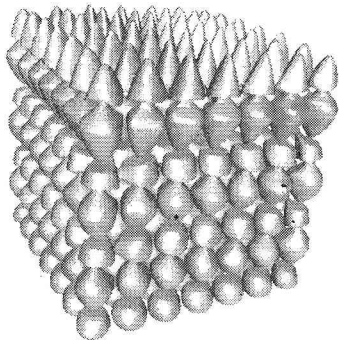


図-16 $t = 200\Delta t$

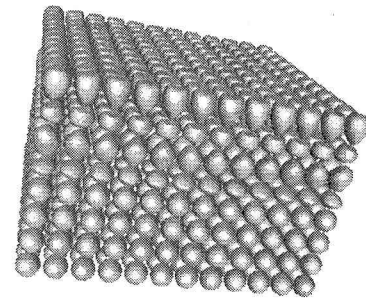


図-19 $t = 60\Delta t$

ていた。

解析に要した実効 CPU 時間の最大値は 10 時間 47 分である。また code 中で確保すべき配列の大きさは 1 プロセスあたり約 19.1GB、全プロセスで約 152.8GB であった。仮に MPI による並列化を行わず 1 プロセスで計算するならば、約 116GB の配列が必要である。

HPC2500 は 1 ノード当たり 512GB 使用できるのでさらなる大規模問題の解析が可能であると言える。

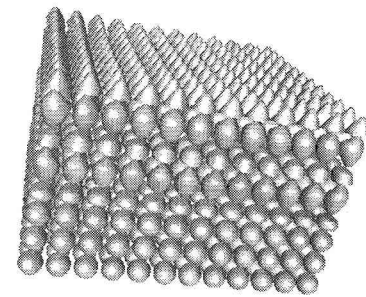


図-20 $t = 80\Delta t$

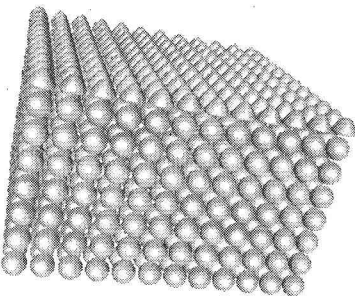


図-17 $t = 20\Delta t$

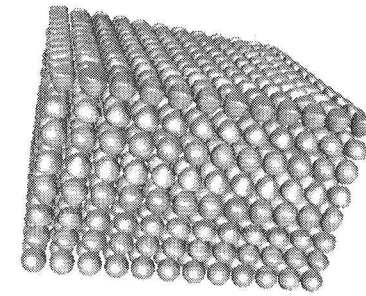


図-21 $t = 100\Delta t$

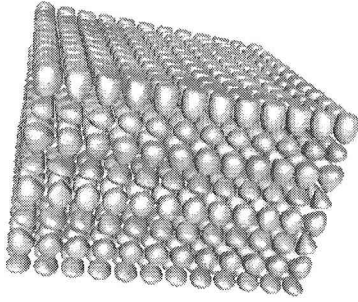


図-22 $t = 120\Delta t$

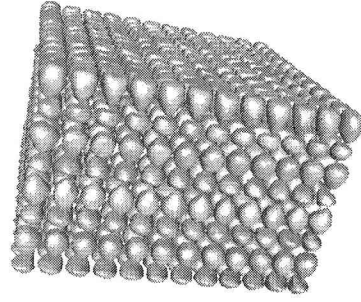


図-25 $t = 180\Delta t$

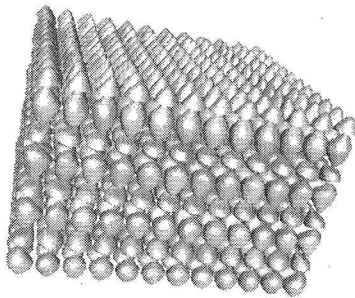


図-23 $t = 140\Delta t$

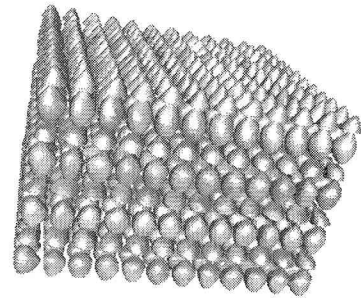


図-26 $t = 200\Delta t$

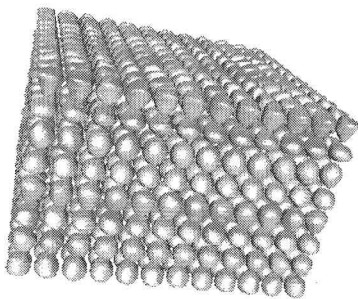


図-24 $t = 160\Delta t$

6. 結言

共有メモリー型計算機において OpenMP を用いることにより、非常に容易に 3 次元動弾性境界要素法の高速解法がスレッド並列化されることを示した。また、その並列化効率を調べ有効性を確認した。

さらに、MPI-OpenMP のハイブリッド並列を行った。それにより問題の大型化を図り、百万を越える自由度の散乱問題を解くことができた。必要メモリー、計算時間の両面から考えても、今後さらに大きい問題を解くことは可能であると考えられる。

参考文献

- 1) Ergin, A. A., Shanker, B. and Michielssen, E.: Fast Evaluation of Three-Dimensional Transient Wave Fields Using Diagonal Translation Operators, J. Comput. Phys., Vol.146, pp.157-180, 1998
- 2) Lu, M., Wang, J., Ergin, A. A. and Michielssen, E.: Fast Evaluation of Two Dimensional Transient Wave Fields, J. Comput. Phys., Vol.158, pp.161-185, 2000
- 3) 高橋徹、西村直志、小林昭一：2次元時間域動弾性問題に対する高速境界積分方程式法、日本機械学会論文集、Vol.67 No.661(A)、pp.1409-1416、2001
- 4) Takahashi, T., Nishimura, N. and Kobayashi, S.: A Fast BIEM for Three-Dimensional Elastodynamics in Time Domain, Eng. Anal. Boundary Elements, Vol.27, pp.491-506, 2003 (see Vol.28, pp.165-180, 2004 for Erratum to "A Fast BIEM for Three-Dimensional Elastodynamics in Time Domain")
- 5) <http://www.openmp.org/>
- 6) <http://www-unix.mcs.anl.gov/mpi/>
- 7) 西村直志、大谷佳広：共有メモリー計算機における境界積分方程式法の並列化について、境界要素法論文集、Vol.20、pp.83-86、2003

(2004年4月16日受付)