

3次元時間域動弾性問題における境界積分方程式法の アルゴリズム改良と並列化

Algorithm improvements and parallel computing of the three dimensional elastodynamic
boundary integral equation method in time domain

Hitoshi YOSHIKAWA*, Naoshi NISHIMURA** and Shoichi KOBAYASHI***

吉川仁・西村直志・小林昭一

*Student member, M. Eng., Dept. Global Environment Eng., Kyoto Univ.
(Yoshida-Honmachi, Sakyo-Ku, Kyoto 606-8501)

**Member, Dr. Eng., Prof., Dept. Civil Eng., Kyoto Univ.
(Yoshida-Honmachi, Sakyo-Ku, Kyoto 606-8501)

***Fellow, Dr. Eng., Prof., Dept. Construction Eng., Fukui Univ. of Technology
(3-6-1 Gakuen, Fukui 910-8505)

Analyses of various real world problems related to elastic wave propagation reduce to large scale problems of three dimensional elastodynamic BIEM (boundary integral equation method) in time domain. Conventional approaches in BIEM, however, are quite expensive in large scale problems in terms of both memory requirements and the CPU time. We therefore propose improved algorithms for 3 dimensional elastodynamic BIEM in time domain and parallelize the code using MPI in order to reduce the memory requirements and the CPU time. The efficiency of the improved code is demonstrated with simple crack problems, as well as with more realistic wave analysis related to crack determination.

Key Words : time domain BIEM, algorithm, parallel computing, MPI, elastodynamics

1. 序論

著者らは、これまでレーザ計測により得られる実データからクラックの位置や形状を決定する問題¹⁾を、時間域境界積分方程式法 (time domain boundary integral equation method) を用いて解析する研究を行なっている^{2),3)}。クラック決定の手順は次のとおりである。まず、未知の表面クラックを有するアルミニウム合金製の供試体に、超音波トランスデューサにより弾性波を発生させ、クラックからの散乱波をレーザ干渉計で計測し実験データを得る。次に、クラックを形状パラメータで記述し、クラックからの散乱波を時間域 BIEM を用いて数値的に求める。最後に、実験値と数値解との差からなるコスト関数の値を最小とする形状パラメータからクラックを決定する。現在実験に用いているアルミニウム合金の S 波速度は約 3100m/sec であり、トランスデューサの中心周波数は 500KHz 程度である。従って、扱う S 波の波長は約 6.2mm となる。そうすると、S 波の挙動を正確に表現するには 1 波長に多くの境界要素を含む必要があることから、境界要素のサイズは一辺 1mm 程度となるので、数 cm 四方の領域を解析するのに数千～数万自由度の分割が必要になる。また、時間ステップ幅も要素サイズに合わせて小さくする必要があるので、数値解析は時間域の大規模問題となる。

時間域の BIEM⁴⁾ では、解が時間に関する畳み込み積分の形で表現されるため、ある時刻における解は、過去の全履歴の影響を受ける。数値解析を行なう上で、各時間ステップにおいて計算される影響係数を全て store する事が可能であれば、積分評価に要する計算時間は $O(N_t)$ (N_t は時間ステップ数) 程度に収まるが、大規模問題を取り扱う場合は必ずしも十分なメモリーが確保出来るとは限らないので、更に計算量が増大する。実際、大規模問題の場合は、ある程度の時間ステップまでしか影響係数を store できないので、メモリーを使い切る時刻以降の解を求めるためには、その時刻以後の全ての時間ステップにおける影響係数を、時間ステップ毎に再計算する必要がある。このような手法に基づいた時間域 BIEM を用いる場合、素朴なアルゴリズムを用いると、積分に要する計算時間はほぼ $O(N_t^2)$ となり、解析には多大な計算時間を要する事になる。

このような時間域 BIEM の問題点を解決するため、本報では、従来型の (多重極法によらない) 時間域 BIEM の計算効率を 2 つの面から改良する事を提案する。まず、畳み込み積分に関するアルゴリズムを改良し、無駄なメモリー使用や再計算を極力排除する方法を検討する。さらに、より多くのメモリーの確保と、計算時間の短縮のために、コードの並列化を検討する。一般に境界積分方程式法は得られる係数行列が密であることから、有限要素法などに比べてかえって単純に並列化で

きるが、時間域の境界積分方程式法では係数行列が疎になる事を考慮した並列化が必要となる。これらの改良により、大規模問題においても実用的な計算時間で解析を行ない得る BIEM コードが得られる事を示す。

なお、BIEM の高速解法の一つである多重極法の 3 次元時間域動弾性問題への適用も研究されているが⁵⁾、解法が複雑であり、まだ改良の余地が残されている。そこで、本報では、従来型の時間域 BIEM のみを取り扱う。

本報の構成は次のとおりである。まず、時間域 BIEM の定式化を示し、畳み込み積分に関するアルゴリズムの改良について述べる。3 次元動弾性問題を数値例として解き、アルゴリズムの改良により計算時間が短縮される事を示す。次に、MPI (message-passing interface)⁶⁾ を用いた時間域 BIEM の並列化について述べる。単一クラックの開口変位を求める問題を複数台のプロセッサで解き、並列化の有効性を示す。最後に、提案する改良された時間域並列 BIEM を実データを用いた超音波のクラックによる散乱問題に適用し、実用上十分な精度で実験値を再現し得る事を示す。

2. 改良された 3 次元動弾性時間域並列 BIEM

2.1 問題の定式化

3 次元弾性体 D に存在する単一クラック S による入射波の散乱問題を考える。例えば、境界条件が全て表面力 t で与えられる場合、次の初期値境界値問題を未知の変位場 u について解く事になる。

$$\begin{aligned} \mu \Delta u + (\lambda + \mu) \nabla \nabla \cdot u &= \rho \ddot{u} \quad \text{in } D \setminus S \times (t > 0), (1) \\ Tu &= t \quad \text{on } \partial D \times (t > 0), \\ u|_{t=0} = \dot{u}|_{t=0} &= 0 \quad \text{in } D, \\ Tu^\pm &= 0 \quad \text{on } S \times (t \geq 0), \\ \varphi &= u^+ - u^- = 0 \quad \text{on } \partial S. \end{aligned}$$

ここに、 λ, μ はラメ定数、 ρ は密度、 T はトラクション作用素、 $+$ ($-$) はクラックの単位法線ベクトルの正(負)からの極限值、 $(\cdot)$ は時間微分、 φ はクラックの開口変位を表す。

式 (1) に対応する境界積分方程式は x が境界 ∂D 上にある場合には次のように書ける。

$$\begin{aligned} \frac{1}{2} u(x, t) &= \int_{\partial D} \Gamma(x, y, t) * Tu(y, t) dS \\ &\quad - \text{v.p.} \int_{\partial D} \Gamma_I(x, y, t) * u(y, t) dS \\ &\quad + \int_S \Gamma_I(x, y, t) * \varphi(y, t) dS, \quad x \in \partial D. (2) \end{aligned}$$

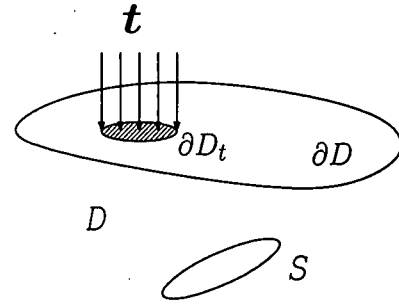


Fig. 1 半無限弾性体内のクラック問題

また、 x がクラック S 上にある場合には次のように書ける。

$$\begin{aligned} 0 &= \int_{\partial D} T\Gamma(x, y, t) * Tu(y, t) dS \\ &\quad - \int_{\partial D} T\Gamma_I(x, y, t) * u(y, t) dS \\ &\quad + \text{p.f.} \int_S T\Gamma_I(x, y, t) * \varphi(y, t) dS, \quad x \in S. (3) \end{aligned}$$

ここに、 $\Gamma(x, y, t)$, $\Gamma_I(x, y, t)$ はそれぞれ動弾性問題の基本解と二重層核、 $*$ は時間に関する畳み込み積分、v.p. は Cauchy の主値、p.f. は発散積分の有限部分を表す。

式 (2), (3) を空間域の内挿関数 N^q 、時間域の内挿関数 M^ℓ を用いて離散化し、次の代数方程式を得る。

$$\begin{aligned} \sum_q A_{ij}^{pq}(\Delta t) \begin{bmatrix} u_j(y^q, n\Delta t) \\ -\varphi_j(y^q, n\Delta t) \end{bmatrix} &+ \begin{bmatrix} \frac{1}{2} u_i(x^p, n\Delta t) \\ 0 \end{bmatrix} \\ &= b_i(x^p, n\Delta t) \\ &- \sum_q \sum_{\ell=1}^{n-1} A_{ij}^{pq}((n+1-\ell)\Delta t) \begin{bmatrix} u_j(y^q, \ell\Delta t) \\ -\varphi_j(y^q, \ell\Delta t) \end{bmatrix}. (4) \end{aligned}$$

ここに、 $x^p \in \partial D$ の時、

$$A_{ij}^{pq}(\ell\Delta t) = \int_{\partial D + S} \int \Gamma_{Iij}(x^p, y, t) M^\ell(t) N^q(y) dt dS_y,$$

$$b_i(x^p, n\Delta t) = \sum_q \sum_{\ell=1}^n Tu(y^q, \ell\Delta t)$$

$$\int_{\partial D} \int \Gamma_{ij}(x^p, y, t) M^{n+1-\ell}(t) N^q(y) dt dS_y,$$

$x^p \in S$ の時、

$$\begin{aligned} A_{ij}^{pq}(\ell\Delta t) &= \int_{\partial D + S} \int (T\Gamma_I)_{ij}(x^p, y, t) M^\ell(t) N^q(y) dt dS_y, \end{aligned}$$

$$b_i(x^p, n\Delta t) = \sum_q \sum_{\ell=1}^n T(Tu)(y^q, \ell\Delta t)$$

$$\int_{\partial D} \int T\Gamma_{ij}(x^p, y, t) M^{n+1-\ell}(t) N^q(y) dt dS_y,$$

であり、 x^p, y^q は、境界要素領域 S_p, S_q の代表点、 Δt は時間ステップ幅である。

2.2 従来の手法

一般に行なわれている BIEM による時間域問題の解析では、各時間ステップ $t = n\Delta t$ ($n = 1, \dots, N_t$) において、時間差が $n\Delta t$ の影響係数 $A_{ij}^{pq}(n\Delta t)$ の積分を計算し、メモリーへ store する。通常、時間域波動問題では、source からの波動が到達しない領域では影響係数は 0 であり、係数行列は疎となる事から、行列の非 0 成分のみを store する。時間差が 1 から $n-1$ までの影響係数 $A_{ij}^{pq}(\ell\Delta t)$ ($\ell = 1, \dots, n-1$) については、以前の時間ステップで既に計算し store されているため、それらをメモリーから呼びだし行列・ベクトル積演算を行なう事で解くべき代数方程式を得る事が出来る。この時、BIEM で最も計算時間を必要とする部分である影響係数の積分計算は各時間ステップで 1 回のみ行なわれ、積分評価に要する全計算時間は $O(N_t)$ となる。しかし、境界要素数や時間ステップ数の大きい大規模問題を取り扱う場合は、全ての影響係数を store だけのメモリーを確保出来るとは限らない。従って、時刻差が $t = m\Delta t$ ($m < N_t$) までの影響係数しか store 出来ない場合、 $\varphi_j(\mathbf{y}^q, n\Delta t)$, ($n > m$) を求める代数方程式を得るには、時刻 $t = n\Delta t$ において影響係数 $A_{ij}^{pq}((m+1)\Delta t), \dots, A_{ij}^{pq}(n\Delta t)$ を計算する必要がある。つまり、影響係数の積分計算を行なうルーチンは $t \leq m\Delta t$ では 1 回、 $t \geq (m+1)\Delta t$ では $n-m$ 回、全体では $\frac{(N_t - m)(N_t - m + 1)}{2} + m$ 回実行される。大規模問題の場合、 m は N_t に比べかなり小さくなり、結局全体の計算量はほぼ $O(N_t^2)$ となって、解析には多大な計算時間を要することになる。

		$\varphi_j(\mathbf{y}^q, \ell\Delta t)$									
$n \setminus$		1	2	3	4	5	6	7	8	9	10
time step	1	①									
	2	②	1								
	3	③	2	1							
	4	④	3	2	1						
	5	⑤	④	3	2	1					
	6	⑥	⑤	④	3	2	1				
	7	⑦	⑥	⑤	④	3	2	1			
	8	⑧	⑦	⑥	⑤	④	3	2	1		
	9	⑨	⑧	⑦	⑥	⑤	④	3	2	1	
	$N_t = 10$	⑩	⑨	⑧	⑦	⑥	⑤	④	3	2	1

Fig. 2 素朴な時間域 BIEM

2.3 時間域 BIEM アルゴリズムの改良

(1) 畳み込み積分に関するアルゴリズムの改良

本節では、大規模 3 次元動弾性問題を実用的な時間で解くために、畳み込み積分に関するアルゴリズムを

改良し、影響係数を計算するルーチンを実行する回数を減らす事を試みる。

時刻 $t = n\Delta t$ において計算される影響係数 $A_{ij}^{pq}(n\Delta t)$ は、時刻 $\ell\Delta t$, ($\ell = n, \dots, N_t$) において $u_j(\mathbf{y}^q, (\ell+1-n)\Delta t)$, $\varphi_j(\mathbf{y}^q, (\ell+1-n)\Delta t)$ (以下、繁雑さを避けるため、代表して φ_j のみ標記する) との行列・ベクトル積演算に用いられ、得られるベクトルは代数方程式の右辺へ加えられる。しかし、時刻 $t = n\Delta t$ において、既に解 $\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($\ell = 1, \dots, n-1$) は既知であるので、 $t = n\Delta t$ 以降の代数方程式の構成に必要な行列・ベクトル積演算 $A_{ij}^{pq}(n\Delta t)\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($\ell = 1, \dots, \min(n-1, N_t+1-n)$) を $t = n\Delta t$ の時点で行ない、それぞれ時刻 $(\ell+n-1)\Delta t$ の代数方程式の右辺に加えておく事 (cast forward) が可能である。この考え方を利用すれば、影響係数 $A_{ij}^{pq}(n\Delta t)$, ($n = [\frac{N_t+3}{2}] \dots, N_t$) (ここに $[a]$ はガウス記号であり、 a を越えない最大の整数を表す) の計算は各時間ステップ $t = n\Delta t$ において 1 度行なうだけで良く、また store する必要もない事がわかる。つまり、全計算ステップ数 N_t の問題において、時間差 $\Delta t, \dots, N_t\Delta t$ の係数行列を全て store する必要はなく、その半分の時間差 $\Delta t, \dots, [\frac{N_t+1}{2}]\Delta t$ の係数行列のみ記憶するだけでよい。しかし、大規模問題の解析では、 $m\Delta t < [\frac{N_t+1}{2}]\Delta t$ となる事が多い。ここに、 m は store 可能な影響係数の最大の時間差である。この場合、i) 影響係数 $A_{ij}^{pq}(n\Delta t)$, ($n = 1, \dots, m$) は時刻 $t = n\Delta t$ において計算し store する。ii) $A_{ij}^{pq}(n\Delta t)$, ($n = m+1, \dots, [\frac{N_t+1}{2}]$) については、 $\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($1 \leq \ell \leq N_t+1-n$) との行列・ベクトル積演算が必要となる。しかし、時刻 $t = n\Delta t$ において計算される $A_{ij}^{pq}(n\Delta t)$ は $\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($1 \leq \ell \leq n-1$) との行列・ベクトル積演算しか実行出来ない。そのため、 $A_{ij}^{pq}(n\Delta t)$ を時刻 $t = \{k(n-1)+1\}\Delta t$, (k は自然数で $k(n-1)+1 \leq N_t$) において計算し、 $\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($((k-1)(n-1)+1 \leq \ell \leq \min(k(n-1), N_t+1-n))$) との行列・ベクトル積演算を行ない、それを時刻 $(\ell+n-1)\Delta t$ の代数方程式の右辺に加える。つまり、 $A_{ij}^{pq}(n\Delta t)$, ($n = m+1, \dots, [\frac{N_t+1}{2}]$) を計算するルーチンは全時間で $\frac{N_t-1}{n-1}$ 回実行される。iii) $A_{ij}^{pq}(n\Delta t)$, ($n = [\frac{N_t+3}{2}], \dots, N_t$) については、時刻 $t = n\Delta t$ において計算し、 $\varphi_j(\mathbf{y}^q, \ell\Delta t)$, ($1 \leq \ell \leq \min(n-1, N_t+1-n)$) との行列・ベクトル積演算を行ない、結果を時刻 $(\ell+n-1)\Delta t$ の代数方程式の右辺に加える。この時、全時間で実行される影響係数の積分計算を行なうルーチンの回数は、 $m + N_t - [\frac{N_t+1}{2}] + \sum_{\ell=m+1}^{[\frac{N_t+1}{2}]} \frac{N_t-1}{\ell-1}$ 回となり、従来の大型問題の解析に比べ

てかなり減らす事が出来る。

例えば、時間ステップ数が $N_t = 10$ 、時間差が $m\Delta t = 3\Delta t$ の影響係数まで store 出来るような問題では、素朴な方法 (Fig. 2) によれば影響係数の積分計算を行な

		$\varphi_j(y^q, \ell\Delta t)$									
$n \backslash$		1	2	3	4	5	6	7	8	9	10
time step	1	①									
	2	②	1								
	3	③	2	1							
	4	④	3	2	1						
	5	⑤	4	3	2	1					
	6	⑥	5	4	3	2	1				
	7	⑦	6	5	④	3	2	1			
	8	⑧	7	6	5	4	3	2	1		
	9	⑨	8	7	6	⑤	4	3	2	1	
	$N_t=10$	⑩	9	8	7	6	5	④	3	2	1

Fig. 3 改良された時間域 BIEM

うルーチンを 31 回呼ぶ必要があるが (図中では○で示した)、提案する方法 (Fig. 3) を用いれば 13 回で済む。

(2) 時間変数の内挿関数に関するアルゴリズムの検討

次に、境界積分方程式の時間変数の離散化に用いられる内挿関数を構成するアルゴリズムについて検討する。本報では、特に区分線形の内挿関数について考察する。

区分線形の内挿関数 M^n は次のように表される。

$$M^n(t) = M_{tri}^n(t) - 2M_{tri}^{n-1}(t) + M_{tri}^{n-2}(t). \quad (5)$$

ここに、

$$M_{tri}^n(t) = \frac{n\Delta t - t}{\Delta t}.$$

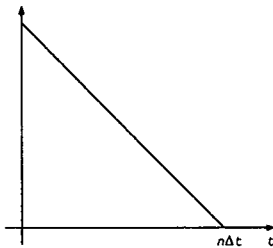


Fig. 4 $M_{tri}^n(t)$

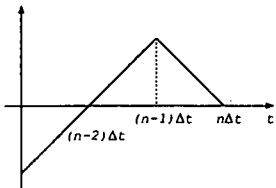


Fig. 5 $M_{tri}^n(t) - 2M_{tri}^{n-1}(t)$

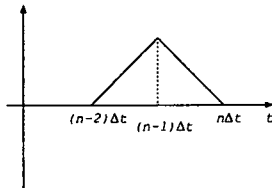


Fig. 6 $M^n(t)$

式 (5) より、時間差 $n\Delta t$ の影響係数 $A_{ij}^{pq}(n\Delta t)$ は次の形で書ける。

$$A_{ij}^{pq}(n\Delta t) = A_{tri,ij}^{pq}(n\Delta t) - 2A_{tri,ij}^{pq}((n-1)\Delta t) + A_{tri,ij}^{pq}((n-2)\Delta t). \quad (6)$$

ここに、

$$A_{tri,ij}^{pq}(n\Delta t) = \int_{\partial D+S} \int \Gamma_{Iij}(x^p, y, t) M_{tri}^n(t) N^q(y) dt dS_y,$$

または、

$$A_{tri,ij}^{pq}(n\Delta t) = \int_{\partial D+S} \int \text{Tr} \Gamma_{Iij}(x^p, y, t) M_{tri}^n(t) N^q(y) dt dS_y.$$

時間に関する積分を解析的に実行する場合、 $A_{ij}^{pq}(n\Delta t)$ の計算は式 (6) を用いて行なうことになる。

式 (6) より各時刻 $n\Delta t$ において $A_{tri,ij}^{pq}((n-1)\Delta t)$, $A_{tri,ij}^{pq}((n-2)\Delta t)$ の情報を持っていれば、 $A_{tri,ij}^{pq}(n\Delta t)$ の計算のみで影響係数を計算できる事がわかる。このとき、時刻 $t = n\Delta t$ において行なわれる作業は以下の通りである。まず $A_{tri,ij}^{pq}(n\Delta t)$ を計算し、store されている $A_{tri,ij}^{pq}((n-2)\Delta t)$, $A_{tri,ij}^{pq}((n-1)\Delta t)$ 、及び、式 (6) を用いて影響係数を計算する。次の時刻 $t = (n+1)\Delta t$ において $A_{tri,ij}^{pq}((n-2)\Delta t)$ はもはや必要ないのでメモリーから解放し、計算した $A_{tri,ij}^{pq}(n\Delta t)$ を store する。この手法により影響係数の計算時間を、 $A_{tri,ij}^{pq}(n\Delta t)$ を store しない手法に比べ約 $\frac{1}{3}$ に短縮できる。しかし、 $A_{tri,ij}^{pq}((n-1)\Delta t)$, $A_{tri,ij}^{pq}((n-2)\Delta t)$ を store するためのメモリーを新たに確保する必要がある。このため、小さい時間差の影響係数の store に使い得るメモリーが減り、取り得る最大の m が小さくなってしまう。一方、2.3 (1) の議論からわかるように、改良された時間域 BIEM では、store 可能な影響係数の時間差 m が小さい程影響係数の計算を行なうルーチンと呼ぶ回数は増える。このように $A_{tri,ij}^{pq}$ を N_t まで store する算法では、大きい時間差の影響係数の計算時間と、小さい時間差の影響係数の store に要するメモリーが trade off の関係となり、このような算法が有効な問題の見極めを慎重に行なう必要がある。例えば、単一クラック問題のように境界の寸法が比較的小さい問題では、ある程度の時間が経過すると source からの波動が境界を通り過ぎてしまう事から、境界要素間の影響はなくなる。このため、 $A_{tri,ij}^{pq}$ の非 0 成分の store にメモリーはそれほど多く必要とならず、 $A_{tri,ij}^{pq}$ を N_t まで store する算法が有効であると考えられる。一方、半無限の領域を持つ問題では、source からの波動の影響範囲に含まれる選点数は時間の経過と共に増大するため、 $A_{tri,ij}^{pq}$ の store に要するメモリーも時間とともに増大し、 $A_{tri,ij}^{pq}$ の N_t までの保持は必ずしも有効ではないと予想される。そこで、以上の考察を確かめるために次節では単一クラック問題と半無限領域の初期値境界値問題を $A_{tri,ij}^{pq}$ を N_t まで store する手法と m までしか store しない手法で解き、問題によりどちらの手法が適しているかを検討する。

(3) 数値例

無限弾性体 D に存在するクラック S による弾性波動散乱問題を考える。考えるクラックは、半径 a の円形クラックである (以下、幾何学量を a で正規化し、 $a = 1$ とする)。入射波はクラックの法線方向 (x_3 方向に一致) に進行する平面 P 波で $\sigma_{33} = p$ は一定とする。Poisson 比 $\nu = 0.25$ 、 $\Delta t = 0.09/c_T$ 、($c_T = \sqrt{\frac{\mu}{\rho}}$)、 $N_t = 11$ とし、クラックの開口変位を求める。なお、境界要素数は 2680 で、未知数は 8040 である。使用した計算機は Alpha EV67 (666Mhz) を CPU とする Compaq 互換機である。なお、使用可能メモリーは 700MB である。

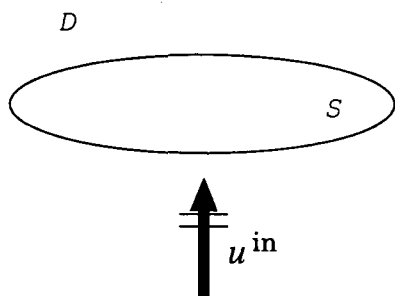


Fig. 7 無限領域内のクラック散乱問題

まず、Fig.2 に示した素朴な時間域 BIEM による解析は 46 分 58 秒の CPU 時間を要したのに対し、改良された BIEM で $A_{\text{tri}ij}^{pq}$ の store を N_t まで行なう手法を用いた解析では 14 分 9 秒で計算が終了した。改良された BIEM で $A_{\text{tri}ij}^{pq}$ の store を m まで行なう手法では解析に 19 分 51 秒要した。この結果より、従来法に比べ改良された時間域 BIEM による解析の方が計算時間が大幅に短縮されている事がわかる。また、 $N_t = 30$ まで計算したところ、改良された BIEM のうち $A_{\text{tri}ij}^{pq}$ の store を N_t まで行なう手法では 71 分 25 秒要し、 $A_{\text{tri}ij}^{pq}$ の store を m まで行なう手法では 84 分 12 秒要した。なお、store 可能な影響係数の時間差 m は、素朴な BIEM と $A_{\text{tri}ij}^{pq}$ を N_t まで store する手法で $m = 3$ 、 $A_{\text{tri}ij}^{pq}$ を m まで store する手法で $m = 5$ となった。

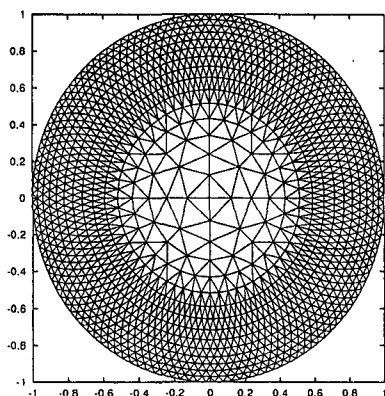


Fig. 8 クラックのメッシュ分割

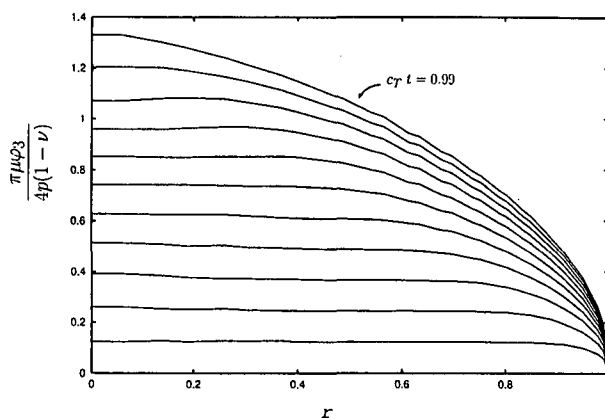


Fig. 9 クラックの開口変位

次に、 x_1x_2 平面を境界とする半無限領域 D で、境界 ∂D 上の半径 a の円形領域 ∂D_t に、荷重 $p(t)$ が作用し、その他の境界では表面力が 0 である問題を解く。ここでも幾何学量を a で正規化し、 $a = 1$ とする。

$$Tu = -p(t)n \quad \text{on } \partial D_t,$$

$$p(t) = \begin{cases} \frac{t}{\Delta t} & (0 \leq t < \Delta t), \\ 2 - \frac{t}{\Delta t} & (\Delta t \leq t \leq 2\Delta t), \\ 0 & (t \geq 2\Delta t). \end{cases}$$

$\Delta t = 0.09/c_T$ 、 $N_t = 30$ とし、 ∂D_t の中心から 3 離れた点での変位を求めた。なお、境界の要素分割は、時間内に ∂D_t からの波動が到達する範囲にのみ行なった。境界要素数は 2808 で、未知数は 8424 である。使用した計算機はクラック問題の解析に用いたものと同じである。

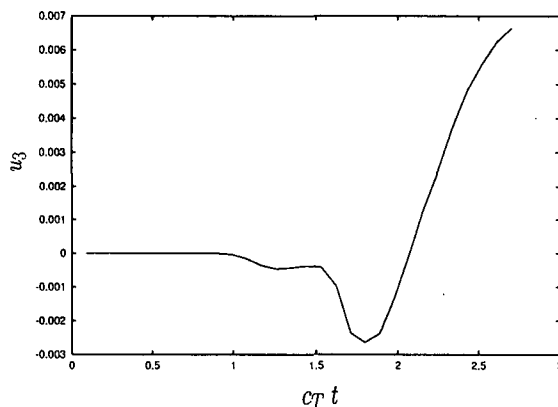


Fig. 10 ∂D_t の中心から 3 離れた点での変位

$A_{\text{tri}ij}^{pq}$ を N_t まで store する手法では $m = 5$ となり、計算時間は 27 分 2 秒要した。なお、 $A_{\text{tri}ij}^{pq}$ を計算するルーチンは 87 回実行され、 $A_{\text{tri}ij}^{pq}(n\Delta t)$ ($n = m+1, \dots, [\frac{N_t+1}{2}]$) を計算するルーチンが同一の n に対して複数回実行されている。一方 $A_{\text{tri}ij}^{pq}$ を m まで store する手法では $m = 15$ となり N_t の半分まで store 可能で、計算時間は 26 分 11 秒であった。 $A_{\text{tri}ij}^{pq}$ を計算

するルーチンの実行回数は60回であるが、非0成分の多い $A_{\text{tri}ij}^{pq}(n\Delta t)$ ($n = m+1, \dots, N_t$) が同一の n に対して複数回実行されているため、計算時間は $A_{\text{tri}ij}^{pq}$ を N_t まで store する手法と余り変わらない。しかし、より大きな時間ステップ数 N_t までの計算が必要な場合、 $A_{\text{tri}ij}^{pq}$ を N_t まで store する事自体が不可能となる。

これらの解析結果から、クラック問題等の有限寸法の境界を持ち、境界上の任意の点から発生した波動が早い時間で境界を通過するような場合は、 $A_{\text{tri}ij}^{pq}$ を N_t まで store する手法を用い、半無限問題や境界のサイズの大きな問題等では、 $A_{\text{tri}ij}^{pq}$ を m まで store する手法を用いた方がより効果的であると言えよう。

2.4 並列化

2.3 での検討により使用出来るメモリーを増やし store する影響係数を増やすことが計算時間の短縮に繋がる事は明らかである。そこで、各時間ステップで必要となる係数行列 A_{ij}^{pq} や $A_{\text{tri}ij}^{pq}$ の計算、記憶、代数方程式右辺の行列・ベクトル積演算を複数台のプロセッサで分割し実行する。全体の計算時間は、複数台のプロセッサに計算を分割する事により短縮され、また使用可能なメモリーが増える事によりさらに短縮される。行列の分割方法には主にブロック分割とサイクリック分割の2つがあるが、時間域問題の係数行列のような疎行列を分割する場合はロードバランスが均等になる分割方法を選ぶ事が望ましい。

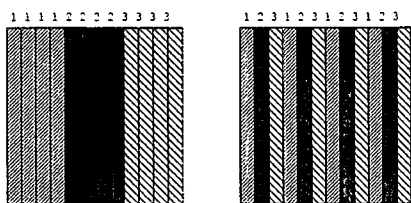


Fig. 11 ブロック分割 サイクリック分割

そこで、2.3 (3) のクラック問題の解析において、3台のプロセッサでブロック分割とサイクリック分割を行ない各プロセッサで計算される影響係数の数を時間ステップごとに示し、ロードバランスを比較する (表1、表2)。分割は source 側の要素を各プロセッサに割り当てる縦方向分割とする。また、Fig. 12、Fig. 13 にブロック分割、サイクリック分割で各プロセッサに割り振られる境界要素を示す。

表 1. 各プロセッサで計算される影響係数の数 (ブロック分割)

time step	processor 1	processor 2	processor 3
1	71520	93513	72019
2	245190	268028	202390
3	466592	429122	357418
4	666335	544069	476728
5	847171	630644	580693
6	1017320	721370	671822
7	1196894	844792	767262
8	1384997	995058	880861
9	1541689	1199256	1029211
10	1586544	1399952	1228552
11	1478142	1534865	1417781

表 2. 各プロセッサで計算される影響係数の数 (サイクリック分割)

time step	processor 1	processor 2	processor 3
1	79012	79023	79017
2	238643	238645	238320
3	417809	417690	417633
4	561793	561210	564129
5	684435	683112	690961
6	801473	799266	809773
7	934411	932084	942453
8	1085914	1083744	1091258
9	1256441	1254467	1259248
10	1405458	1403699	1405891
11	1477794	1476176	1476818

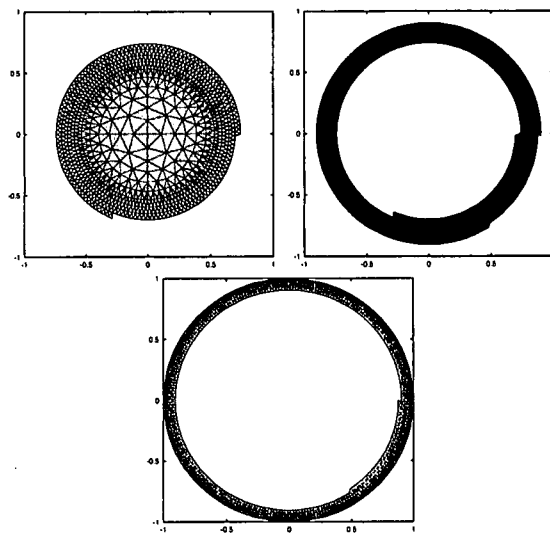


Fig. 12 各プロセッサに割り振られる境界要素 (ブロック分割)

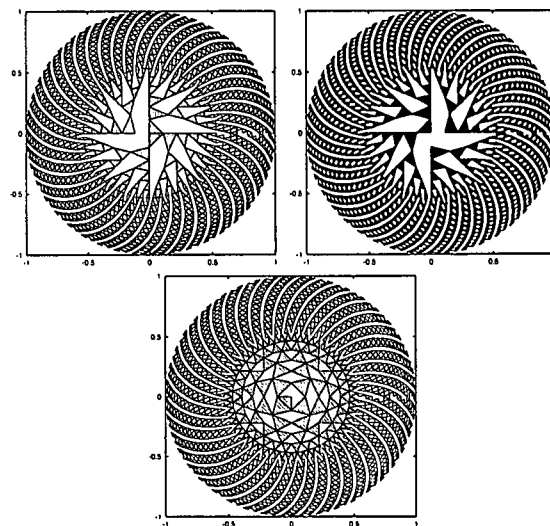


Fig. 13 各プロセッサに割り振られる境界要素 (サイクリック分割)

表1、表2から、本報で取り扱う問題に関してはサイクリック分割が適していると言える。これは、クラックの中央付近の要素とティップ付近の要素とでは、source からの波動の影響範囲に含まれる境界上の選点数が異なるためである。

時間域の境界積分方程式法では、係数行列はバンド型になり、代数方程式の求解には多くの計算時間を要しないため、代数方程式の求解は1台のプロセッサで行なう。その際、長さが未知数程度のベクトルの通信が必要となるが、時間域 BIEM では元来未知数が多くないのでデータ通信量は少なくて済む。解法には繰り返し解法である GMRES を用いる。本報で用いた時間域 BIEM の並列化のアルゴリズムを Fig. 14 に示す。また、1 台から 8 台のプロセッサで並列計算を行ない、費やした計算時間を Fig. 15 に示す。並列化により計算時間がほぼ台数分の 1 となっている事がわかる。なお、使用した計算機は全て CPU が Alpha EV67 (666Mhz)、メモリー 1GB のスペックを持つ Compaq 互換機である。ネットワークは Myrinet を使用し、並列化のソフトウェアは mpich 1.2.4 を用いた。

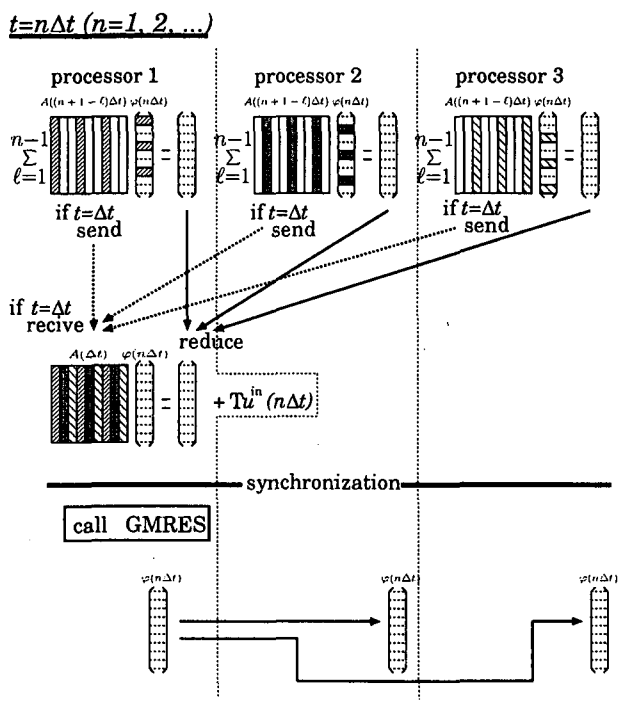


Fig. 14 時間域 BIEM 並列化アルゴリズム

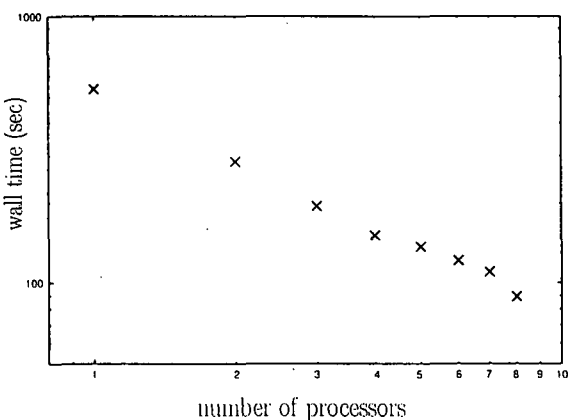


Fig. 15 プロセッサ数と計算時間

3. 大規模動弾性問題の並列解法

前節で示した並列時間域動弾性 BIEM を用いて、アルミ合金製の供試体内に存在するスリット (Fig. 16) による散乱問題を解く。

境界要素数 10316 (未知数 30948)、時間ステップ数 130 の大規模問題を取り扱うため、解析には 8 台のプロセッサを使用する。問題は半無限境界を持つ大規模問題ゆえ、解析には改良された時間域 BIEM を用い $A_{tri_{ij}}^{pq}$ の store を m まで行なう。レーザ干渉計により計測されたデータから復元した入射波 $p_1(t), p_2(t)$ (Fig. 17) を境界条件

$$Tv = -\dot{p}_j(t)n \quad \text{on } \partial D_{t_j} \quad (j = 1, 2)$$

とし (ここに ∂D_{t_1} は中心がトランスデューサの中心と一致する半径 2.72mm の円形境界、 ∂D_{t_2} は供試体とトランスデューサとの接触面から ∂D_{t_1} を引いた残りの境界である)、スリット近傍 (Fig. 16 の×印) の速度応答を計算した。ここで、P 波速度 $c_L = 6180(\text{m/sec})$ 、S 波速度 $c_T = 3180(\text{m/sec})$ 、 $\Delta t = 0.075 (\mu \text{ sec})$ とした。なお、0.2mm 幅のスリットをクラックと見なしクラック問題を解く事も可能であるが、本報ではスリット幅を考慮した解析を行なった。

解析に要したは計算時間は 12 時間 50 分 1 秒であり、影響係数は時間差が $18\Delta t$ のものまで store 可能であった。トランスデューサの中心から 26mm、及び 30mm 離れた点の応答に関して、得られた数値解と計測値を Fig. 18 に示す。距離 26mm の場合の後半部分に計算値と計測値との差が見られるが、これらは概ね良く一致していると言える。

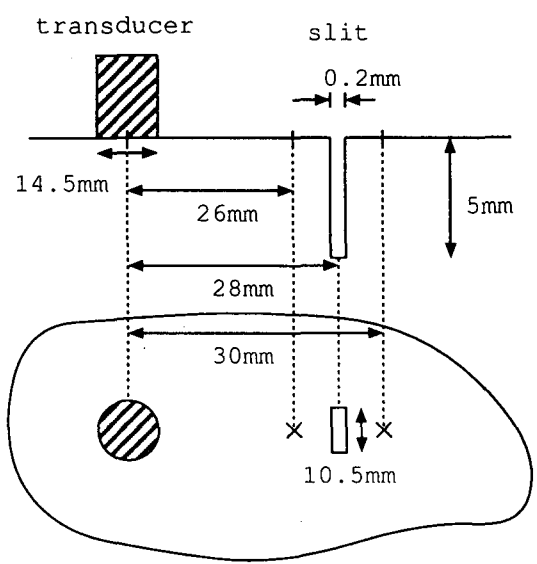


Fig. 16 計測設定

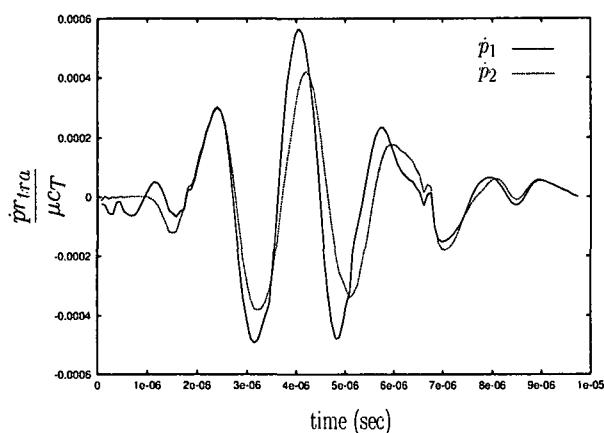


Fig. 17 $\dot{p}_1(t), \dot{p}_2(t)$

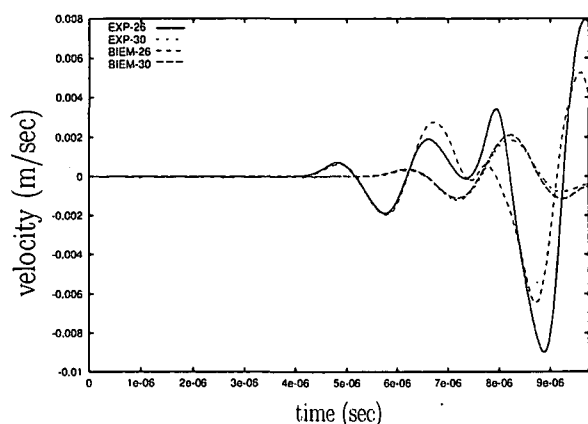


Fig. 18 スリット近傍の速度応答 (BIEM: 数値解, EXP: 計測値)

4. 結論

時間域 BIEM の畳み込み積分に関するアルゴリズムの改良により、各時間ステップにおける影響係数を全て記憶出来ないような大規模問題を実用的な時間で解く事が可能となった。また、本報では時間方向を区分線形関数により離散化した場合のアルゴリズムについて、対象となる問題に応じて影響係数を優先的に store

する手法と、影響係数の計算に必要となる行列を優先的に store する手法のいずれが効果的であるかを検討した。なお、区分一定の内挿関数やその他の内挿関数を用いた場合についても同様の議論を行う事が出来る。

また、BIEM では影響係数の計算に全体の計算時間の殆んどを費すため、並列化は影響係数の計算部分にのみ施し、代数方程式の求解は単一のプロセッサで行った。さらに、プロセッサへの要素の割り当て方を検討し、サイクリック分割が有利である事がわかった。

本報で提案した改良された時間域並列 BIEM を用いれば、これまで BIEM では敬遠されてきたような大規模時間域問題の解析が可能となる。しかし、計算時間のオーダーは改善されていないため、時間ステップが数千、数万の問題になると、問題によっては多大な計算時間を要する事もあり得る。今後は時間域の BIEM の高速解法である PWTD(plane wave time domain) アルゴリズム⁵⁾のような計算時間のオーダーを減らすアルゴリズムに関する研究が必要となるものと考えられる。

参考文献

- 1) N. Nishimura, Crack determination problems, *Theoretical and Applied Mechanics*, 46, (Eds. G. Yagawa and C. Miki), Hokusensha Publ., Tokyo, pp.39-57, 1997.
- 2) 吉川仁、西村直志、小林昭一、レーザ計測を用いた欠陥決定問題のためのトランスデューサ特性同定解析と並列 BIEM による検証, 境界要素法論文集 vol.17, pp.59-64, 2000.
- 3) H. Yoshikawa, N. Nishimura and S. Kobayashi, On the determination of ultrasonic waves emitted from transducers using laser measurements with applications to defect determination problems, 土木学会応用力学論文集, vol.4, pp.145-152, 2001.
- 4) 小林昭一 他: 波動解析と境界要素法, 京都大学学術出版会, 2000.
- 5) T. Takahashi, N. Nishimura and S. Kobayashi, A fast BIEM for three dimensional elastodynamics in time domain, to appear in *Engineering Analysis with Boundary Elements*.
- 6) 青山幸也: 並列プログラミング虎の巻 MPI 版, 日本 IBM 株式会社, 1999.

(2002 年 4 月 19 日 受付)