疎密並列結合・複合化計算機システムによる汎用大規模鍛造シミュレータの開発

岡山県立大学　情報工学部　情報システム工学科

教授　加藤　隆

(平成6年度研究開発助成　AF－94027)

1. 研究の背景

冷間鍛造による機械的性質の優れた高性能･高機能部品の多量一貫生産は，先進工業技術の主要課題の一つとして重大責務[1)]を担っている．近年の高機能部品ニーズの多様化により，冷間鍛造の製造工程設計についても，多品種少量生産および製品サイクルの短命化の急進行によって，塑性加工分野においても従来の実験と経験を主体にした工程設計方法から数値シミュレーション技術を主体にした工程設計へ全面的にシフトする傾向にある．そのため，複雑形状製品の塑性加工に対する全工程のシミュレーションを可能にするだけでなく，最終製品の仕上り精度までを計算予測させるような高度な先進技術が要求され始めている．

しかし，塑性加工向きの剛塑性有限要素解析法に基づく鍛造シミュレーション技術によって，実際的な製品形状や実用金型工具構造をそのまま組込んで高細精度な解析を実現する場合には，解析規模が大規模化することが避けられない．そのため，大規模解析処理を工程設計に最大限に利用するには，大規模鍛造シミュレーションの超高速演算処理化を実現するような高度技術の開発が必要である．

本研究は，こうした超高速大規模変形シミュレーションの実現要求に対して，最新の超高速ネットワークと並列分散処理技法を導入し，疎密並列結合・複合化計算機システム上で稼働する新たな汎用鍛造シミュレータの開発を目指したものである．

2. 変形解析コードのプログラム構造分析

本研究では，筆者を中心にした研究グループで開発した汎用冷温間鍛造シミュレータをベースにして，これに並列処理技術を適用することで鍛造シミュレーションに必要な数値計算全体の高速演算化を図ることとした．

その第１ステップとして，剛塑性有限要素法を用いた鍛造数値シミュレータが，実際にシミュレーションを実行している稼動状態の動的な動作特性を調べることとした．この分析によって大規模解析で最もCPU時間を要する処理部分を抽出し，それらの主要部分に対して並列化技術を主体とした高速化を図ることによって，シミュレータ全体の効率的な高速化を実現することを目指している．

まず，基本的な加工形式に対する数値解析実験を行い，その計算結果からシミュレータの主要構成サブルーチン群の演算実行時のプログラムプロファイル(実行形式オブジェクトコードの演算実行挙動の履歴)を詳細に調べることにした．そのため，ワークステーション(以下EWSと略記)用のUNIX-OSが提供する各種プログラム解析用ユーティリティソフトウェアやコンパイル言語用オプションソフトウェアを多用して以下のようなプログラム構造解析を行った．

2.1 二次元軸対称モデルによる解析

二次元軸対称モデルを対象にしたシミュレータの構造解析については，冷間鍛造を代表する変形様式の基準として円柱素材の据込み圧縮加工過程のシミュレーション中の動的な解析を行った．

解析実行中に節点が新たに工具へ接触を開始するなどの不規則挙動の判定などを始めとして，解析上で境界条件の切替･変更などを含ませないように配慮し，できる限り演算プロファイル内のデータ個々の相互比較が忠実かつ合理的に行われるように心がけた．

演算プロファイルの集計データを整理して，各ルーチンの相対的な演算時間占有比率をまとめた結果が図１である．なお，この相対時間占有率の算出には，個々のシミュレーション計算実行時の各10ステップ分の累積値毎に求めた平均的なプロファイルデータを採用している．

本図から，計算に要する実質時間の大半が3つのシミュレータ内ルーチンsolve, set_A, set_bに費やされていることが明らかである．そして，総要素数(総節点数)の増加に伴って剛性方程式の求解処理を行うsolveの占める割合が徐々に増大し，n=60×60あたりになるとその値が80%台に入り始め，n=100×100に至るとほぼ90%までをsolve自身が占有して他の比率値を完全に圧倒するような結果が得られる．

2.2 三次元モデルによる解析

三次元モデルを用いたシミュレータの構造解析では，三次元剛塑性有限要素法解析において最も典型的な単純形状

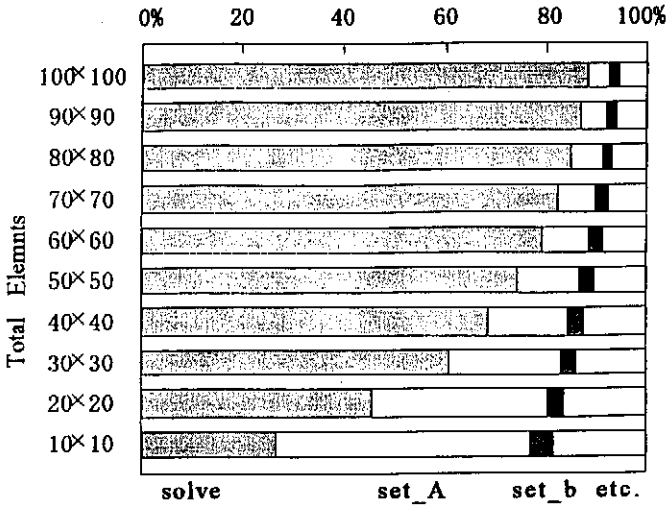

図1　軸対称モデル解析における主要ルーチンの時間占有率

表1　評価用FEMモデルの一覧

モデル名	要素数	節点数	自由度	平均帯幅
m3D04	4^3	5^3	375	76
m3D05	5^3	6^3	648	110
m3D06	6^3	7^3	1,029	149
m3D07	7^3	8^3	1,536	194
m3D08	8^3	9^3	2,187	245
m3D09	9^3	10^3	3,000	302
m3D10	10^3	11^3	3,993	365
m3D11	11^3	12^3	5,184	434
m3D12	12^3	13^3	6,591	509

を想定し，四角柱を評価対象にして数値解析を行った．要素分割は四角柱各辺の分割数が等しくなるように行った．図2は解析結果の一例を示す．これは底面と手前に見えている2つの側面を対称面とし，上面の1/4区画に下向きに強制変位を与えた場合の塑性変形解析の結果を示したものである．

本研究では，表1に示すように各辺の分割数Nを広範囲に設定した異なる規模の解析モデルを用意した．各モデル名はm3Dが三次元モデルであることを示し，それに続く数字が各辺の分割数を表している．

各モデルについて1プロセッサのみ稼動させた場合の数値解析を実施し，サブルーチンごとの処理時間占有率を求めた結果が図3である．ここでdecompは剛性マトリックスのコレスキー分解，set_Aは剛性マトリックスの作成を行うルーチンである．

図のように，これら2つのルーチンが演算時間の大部分を占めており，解析モデルが大規模になるほどdecompが支配的になっている様子がわかる．例えばこの解析条件の中で最大規模のm3D12では，decompの時間占有率が95%となっている．

高速化が要求される程度は，必然的に大規模モデルになればなる程強まることは自明であるが，そうした条件になるほどdecompのみを並列化して高速演算化するだけでも十分な効果が得られるものと予想される．

2.3 変形解析コードのプログラム構造分析結果

前節までの解析結果によって，最新鋭のEWS上で許容最大限までの大規模な鍛造シミュレーションを行わせることで，その総計算時間長とその際の演算プロファイル情報が得られた．その結果，実製品相当の複雑な形状を対象に行

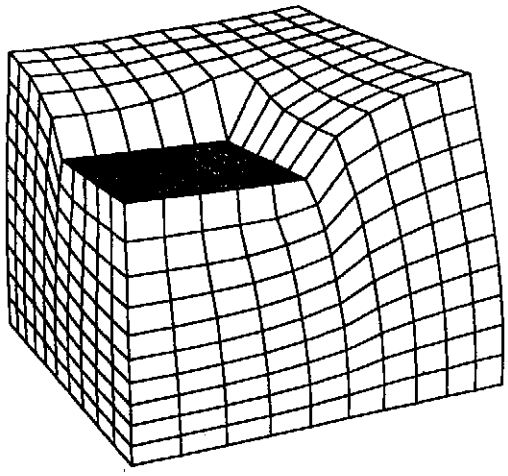

図2　三次元モデル解析例（30%圧下後）

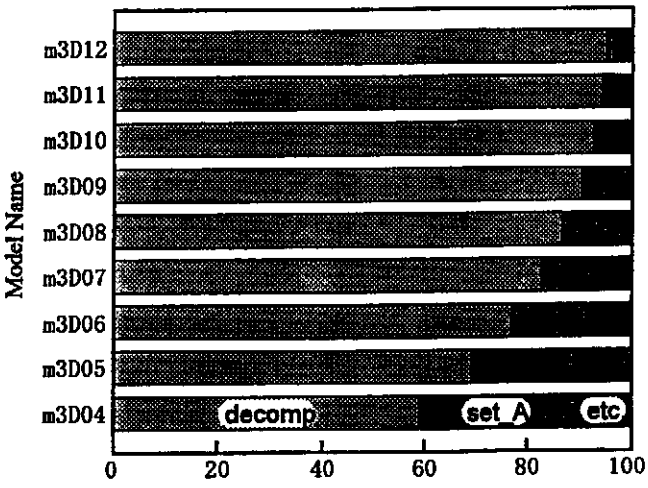

図3　三次元モデル解析における主要ルーチンの時間占有率（%）

われるような比較的大規模な解析条件になると，その計算実行時間の大部分が剛性マトリックス方程式の求解に費やされることが明らかになった．これによって，鍛造シミュレータ全体の計算実行時のパフォーマンス向上を図るためには，剛性方程式の求解部分の高速化に全精力をつぎ込めば良く，逆にその他の部分は，注力して高速化を図ってもシミュレーション全体としては目立った効果を得るのが難しいことがわかった．

3. 剛塑性有限要素法の並列化

前出の第２章で述べたように，大規模な有限要素解析では，剛性方程式の求解が処理時間の大半を占める傾向にある．したがって，この部分を並列処理によって集中的に高速化することにより，全体の処理時間の大幅な短縮化を図ることにした．

本研究では，並列処理による高速化を実現するため，まず，並列処理コンピュータとして最も一般的で汎用性の高い密結合型並列コンピュータの使用を考えて，その前提条件となる共有メモリ方式のプログラミング技術に基づく直接法の並列化を試みた．これによって剛性方程式全体，すなわち連立方程式の高速求解を実現することにした．

実計算に当たっては，密結合型並列コンピュータの商用機である(米)シーケント社製のSymmetry-2000システムを導入し，これを実際に使用して有限要素解析法で頻出する正定値対称行列に対するコレスキー分解の並列処理化を最優先にして検討した．

3.1 コレスキー分解の並列実行の可能性

本研究では，次式で示すコレスキー分解にタイリング[2]およびブロック化を施した上で，これを密結合型並列コンピュータSymmetry-2000システム上で並列処理する方法を検討した．

$$A = CC^t \tag{1}$$

$$c_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} c_{ki}^2} \tag{2}$$

$$c_{ij} = \frac{1}{c_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} c_{ki} c_{kj} \right), \quad (i < j) \tag{3}$$

コレスキー分解の一連の式を見ると，第i行に位置する各要素の分解を行う際に（このとき第i-1行までの全要素の分解計算は終了している），対角要素の分解が終了した段階で，同じ行の第i+1列以降に位置する各要素の分解が可能となり，しかもこれらの計算には相互に依存関係がないので全く独立して実行できることがわかる．したがってこの部分には並列処理の適用が可能となる．

そして，このことはブロック化を施した場合でも同様なことがいえる．すなわち，第iブロック行の分解計算を行う際，対角ブロックの分解計算が終了した時点で，同じブロック行の第i+1ブロック列以降の各ブロックの分解が可能となり，しかもそれぞれを並列に実行することができる．

3.5 プロセッサ間の同期と分解計算の割当

本研究では，並列動作するプロセッサ間の同期機構としてイベントカウント[3]を使用した．本研究ではイベントカウントEの値を，分解が完了した対角ブロックの最新の位置情報に対応させた[4]．

各プロセスはEの値を調べることにより，ピボットブロック列の対角ブロックが分解済みであるかどうか，すなわち分解計算の際に参照するピボットブロック列が準備できているかどうかを知ることができる．各プロセッサが分解計算を担当する範囲はブロック列単位で周期的(cyclic)かつ静的に割り当てる．つまり同じブロック列位置の各ブロックの分解は，同一のプロセッサが処理する．

図４は各プロセッサへの割り当てと処理手順を示したものである．ここではプロセッサ数を３とし，それぞれ P0, P1, P2で表す．対角ブロックの分解はi番目まで完了していると仮定する．このときイベントカウントの値はE=iとなっている．

この図に示されるように，プロセッサP0は第iブロック行中の担当ブロック 1^{P0}, 2^{P0}を分解すると，次のブロック行に移る前に AWAIT(E,i+1)を実行し，i+1番目の対角ブロック2^{P1}の分解の完了を待つ．完了が確認されると第i+1ブロック行の担当ブロック3^{P0}, 4^{P0}, 5^{P0}を分解する．次にi+2番目の対角ブロック5^{P2}の分解の完了を待つ．完了が確認されるとまず6^{P0}を分解する．6^{P0}の直下ブロック7^{P0}は対角ブロックなのでこれを次に分解し，これが終了した時点で ADVANCE(E) を実行してイベントカウントEを増加させる．その後，第i+2ブロック行の残りのブロック8^{P0}, 9^{P0}を分解する．

プロセッサP1は，1^{P1}を分解した後，この直下のブロック2^{P1}が対角ブロックなのでこれを先に分解し，ADVANCE(E) を実行する．続けて3^{P1}, 4^{P1}, 5^{P1}, 6^{P1}を分解する．5^{P1}の分解に移る際にはAWAIT()は必要ない．なぜならこのブロック行の対角ブロックの分解はP1自身がすでに行っているからである． 7^{P1}の分解を行う前には AWAIT(E,i+2) により対角ブロック 5^{P2}の分解終了を確認する．もちろん，プロセッサP2についても同様な手順である．

以上の一連の手順の要点をまとめると次の①～③のようになる．

①分解計算の担当範囲を，ブロック列単位でプロセッサに割り当てる．

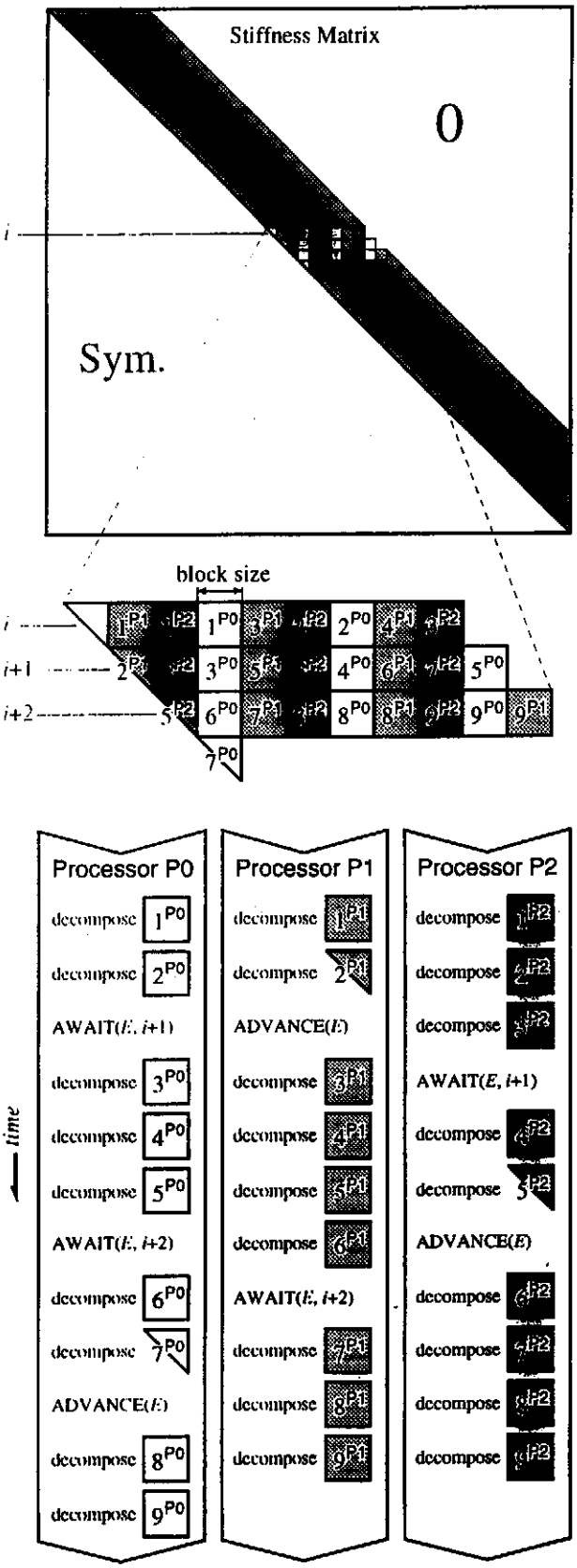

図4　コレスキー分解の並列処理手順

②プロセッサ間の同期にはイベントカウントを使用し，分解が終了した対角ブロックの最新の位置情報を値として保持させる．

③対角ブロックは可能な限り早い段階で分解する．つまり，対角ブロックのすぐ上に位置するブロックの分解後は，同じブロック行の他のブロックを分解する前に直下の対角ブロックを先に分解し，早期にイベントカウントを増加させ他のプロセッサに処理を続行させる機会を与えやすくする．

このように，同じブロック列中の各ブロック，すなわち上下に並んだ各ブロックの分解を同一のプロセッサに割り当てることにより，同期が単純化されていることがわかる．つまり，あるブロックの分解計算を行う時点では，同じブロック列中の上方の各ブロックがすべて分解済みであることが保証される．

ブロック化したコレスキー分解では，あるブロックの分解計算を行う際に，そのブロックの上方にあるすべてのブロックおよびピボットブロック列を参照するが，これらはすべて分解済みでなければならない．ここで示したプロセッサの割り当て方法では，ピボットブロック列の対角ブロックが分解済みであることのみを確認するだけで，この条件が満足される．

3.6 シミュレーションの実行結果

前節までに述べた手法を用いてコレスキー分解を行うサブルーチンdecompの並列化を行い，評価用の解析モデルに対してSymmetry-2000システムの実機上で数値計算を実行し，このときの経過時間を測定して高速化の評価を行った．

図5は本研究で扱った最大規模の評価用モデルm3D12に

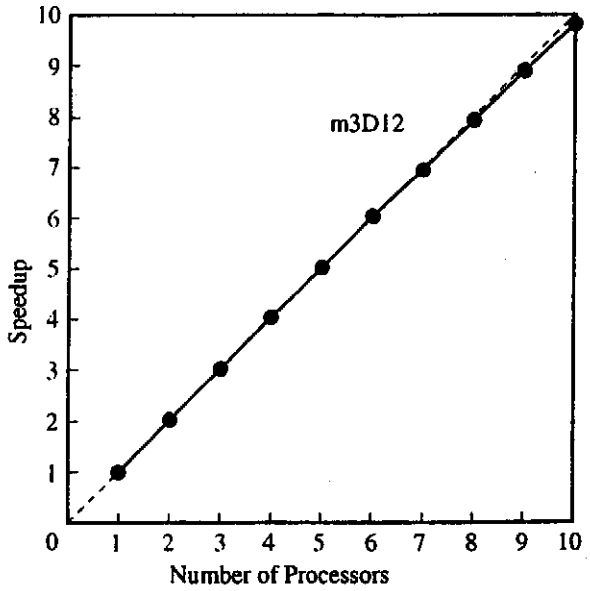

図5　プロセッサ数と高速化率

ついての結果である．この程度の規模になると帯幅は十分に大きく（平均帯幅=509）なるため，10プロセッサの並列稼動時の高速化率は9.8となり，きわめて理想値に近い結果が得られている．

これらの結果から，同期機構としてイベントカウントを使用する手法は，並列処理を用いた剛性方程式の高速求解に非常に有効であるといえる．

4. 超高速ネットワークを用いた並列分散処理

第３章に前述したように，比較的小規模な密結合型並列コンピュータを用いた連立方程式の高速求解においては，イベントカウンタが有効であり，ほぼCPU数に比例した高速化を達成することができた．

密結合型の並列コンピュータは，共有メモリという構造上の特徴によってプログラミング上の取扱が容易であるという特長を有する，しかし，多CPUを装備した大規模並列マシンの製造が困難であり，マシンが高価になるなどの欠点を有する．そのため，本研究が目指している大規模・超高速シミュレーションを実現するためには，多並列CPUによる大規模並列処理の実現が密結合型並列コンピュータに比べて著しく容易な疎結合型並列コンピュータの利用が適していると考えられる．

疎結合型並列コンピュータは，密結合型並列コンピュータに比べてCPU間のデータ通信速度が著しく遅いため，通信路が高速処理におけるボトルネックとなると言われている．そこで，本研究では通信路として超高速ネットワークを用いることにした．超高速ネットワークを用いて相互接続されたワークステーション群（ワークステーションクラスタ）を用いて疎結合型並列処理を実現し，大規模連立方程式の高速求解における有効性を検討した．

超高速ネットワークとしては，現時点で最も実績があり，高通信負荷状態における性能低下が少ない光ケーブル接続方式のFDDIと等価なツイストケーブル接続方式のTPDDIを用いた．

4.1 データ転送速度

TPDDIの仕様上の最大データ転送速度は100Mビット/秒である．しかし，実際のデータ通信においては，制御情報の交換等の諸原因によって，データ転送速度は100Mビット/秒以下になると予想される．このため，実際のワークステーションクラスタを構成するマシンを用いて実用データ転送速度を計測した．

本研究で用いたワークステーション用OSであるUNIX-OS上での通信方式として，最も基本的かつ一般的なソケットを用いた．ソケットを用いることにより，同一プログラムでTPDDI，イーサネット，主記憶の３種類の異なった媒体を用いた通信が可能である．ここで，主記憶とは同一マシン内で２つのプロセスがソケットを用いて１対１でデータ交換することを意味する．

これら３つの媒体を用いた場合のデータ転送速度を一回の通信動作で送受信可能なデータサイズ（セグメントサイズ）を変えて実測した結果を図６に示す．各グラフは，それぞれの媒体を用いた場合に一回の通信で送受信可能な最大データサイズまでを計測した結果を示している．

図６から明らかなように，イーサネットが最も低速であり，TPDDI，主記憶の順で高速になっている．しかし，その速度差はセグメントサイズが小さいくなるに従って減少し，各通信方式間で有意差が認め難い程に接近してくる．こうした実測結果から，高速通信路を採用してその効果を充分発揮させるためには，セグメントサイズを極力最大値に近づけるような通信容量を常時維持することが肝要であるとがわかる．

4.2 反復法

密結合型並列コンピュータを用いた連立方程式の高速求解には，イベントカウンタを用いた並列コレスキー法が非常に有効であった．しかし，実測したTPDDIの通信速度から次のことが明らかになった．すなわち，TPDDIを用いたCPU間のデータ転送であっても，その転送速度は密結合型並列コンピュータが行える共有メモリ上でのデータ共有によるデータ交換に比べれば著しく低速である．

そのため，コレスキー法のような直接法を疎結合型並列コンピュータ上での並列化した場合には，イベントカウン

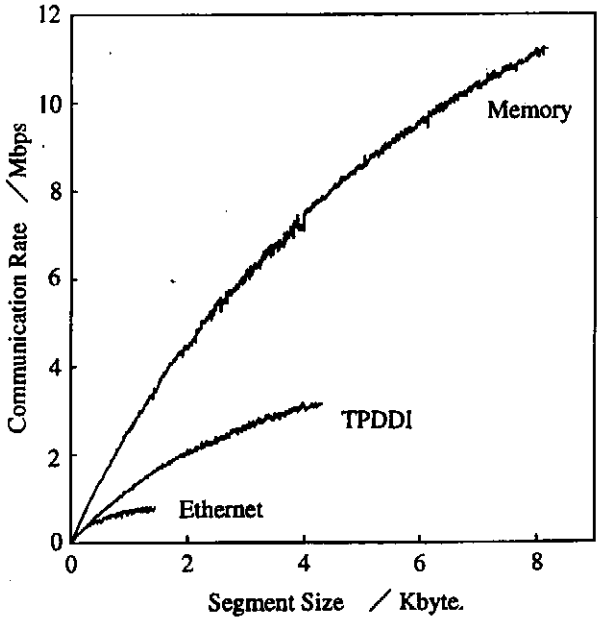

図６　通信速度比較

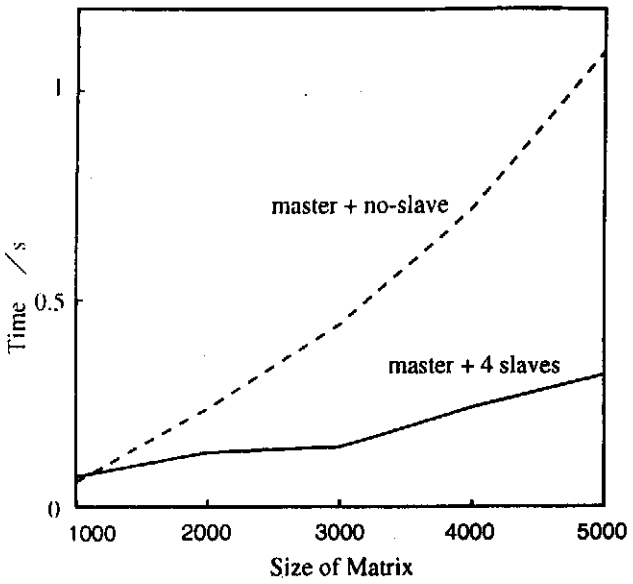

図7　疎結合型並列処理による処理時間短縮

タを用いてCPU間の処理の同期を高速化しても，データ共有実現に必要な通信が全体の処理効率を低下させることになる．

以上のことから，疎結合並列処理を用いて剛性方程式の求解を高速化するには反復法が有利と考えられる．そこで，本研究では反復法の中で最も一般的な共役勾配法[5]を用いることとし，その並列化を行った．

並列化は，共役勾配法のアルゴリズム中で最も計算時間を要する行列とベクトルとの内積部分を並列に実行させることにした．

4.3 疎結合型並列処理

本研究では，TPDDIで相互接続した５台のワークステーション群（マスタ×１，スレーブ×４）から成るワークステーションクラスタを用いた．このワークステーションクラスタ上で並列化した共役勾配法を用い，有限要素法を想定した係数行列の連立方程式の高速求解を試みた．

求解に要した時間と連立方程式の規模との関係を図７に示す．同図から，処理時間が最大約１／３に短縮されていることがわかる．すなわち，４台のスレーブを用いることにより，最大約３倍の高速化を達成できたことになる．

また，高速化の効果は，解くべき方程式の規模が増大するほど高くなっている．これは，方程式が大規模になるほど通信に要する時間に比べて演算時間が長くなり，その結果として並列化による高速化が有効になることを示すものといえる．

5. 結　び

今後の鍛造加工技術の高度化を図るには，大規模な変形解析シミュレーションによる迅速な情報提供と設計支援が必須になると予測される．大規模な変形解析シミュレーションの実用化にとって最重要課題の一つである処理時間の大幅な短縮化を実現するための高速演算化を達成するには，本研究で示したように，まず第一に，並列処理技術に基づく剛性方程式の高速求解技術を開発することが最も効果的であることが明らかである．

さらに，剛性方程式の高速求解の実用手段として，直接法（コレスキー法）を用いる場合は密結合型並列処理が有効であることが分かった．また，疎結合型並列処理が単に大規模解析を扱いやすいだけではなく，最新の超高速ネットワークを用いることにより大規模解析の高速化に有効であることが明らかになった．

しかし，今後に予想される超大規模シミュレーションを実用的な時間内に完了させ得るような超高速化には，密結合と疎結合並列処理手法とを融合化した疎密結合型並列処理が必要である．そのため，本研究成果をもとに，さらに並列計算技術の開発を目指した基礎研究を行っていく必要がある．

謝　辞

本研究の遂行にあたり，(財)天田金属加工機械技術振興財団の研究助成を受けたことを記し，謝意を表します．

参考文献

1) Kudo,H.: Proc.of 25th MTDR Conf.(1985),49. (1985),48.
2) 寒川 光：RISC超高速化プログラミング技法，(1995)，共立出版.
3) Reed D.P. and Kanodia R.K.: Synchronization with eventcounts and sequencers, Comm.ACM. Vol.23, No.2 (1979).
4) 宮川佳夫，松田 章,加藤 隆：並列処理シンポジウムJSPP'96 (1996),25.
5) 藤野清次，張 紹良:反復法の数理,(1996),33,朝倉書店.