並列化ツール PVM による剛塑性有限要素解析の高速化*

松田 章^{*1}, 加藤 隆^{*1}, 宮川 佳夫^{*1}
河部 徹^{*2}, 濱谷 智哉^{*3}Parallel Acceleration of Rigid-Plastic Finite Element Analysis
by Parallelization Tool of PVMAkira MATSUDA, Takashi KATO, Yoshio MIYAKAWA,
Tooru KAWABE and Tomoya HAMATANI

For large-scale forging simulation by rigid plastic finite element method, it is important to reduce or accelerate the calculation time which is mostly occupied with the solver part of stiffness equation. In this paper, we introduce the parallelization tool of PVM (Parallel Virtual Machine) to improve the solver part into our cluster system of workstations. We employ TPDDI network as fast communication path between workstations, so PVM provides two excellent functions of Multi-casting and Direct-linking which enable the faster inter-communication about 50% less consumption of calculation time in comparison to the case of standard procedures on PVM as if the case of traditional socket sequence. From the calculation test for forging simulations, our parallelized solver accelerates the calculation of single workstation about 5.4 times on the system developed by using two PVM's functions above. Then, we conclude that PVM provides a new convenient means to high speed execution of large-scale numerical simulations.

Key Words: Finite Element Method, Numerical Analysis, Computational Mechanics, Parallel Computing, Deformation Simulator, Cluster of Workstations, Parallel Virtual Machine

1. 緒 言

最近の製造現場では、多品種・少量・迅速生産方式が一般化しつつあり、最適な加工条件を実験技法によって求めることは、時間的にもコスト的にも非現実的になりつつある⁽¹⁾。このため、数値シミュレーションによって最適な加工条件を求めるための高細精度解析技術の重要性が高まり、その多機能化と高速化に大きな期待が寄せられている⁽²⁾。しかし、実務レベルで通用するような高細精度解析を実現し、大規模シミュレーションを実施可能にするには、数値シミュレーションに要する膨大な計算時間を大幅に短縮する必要がある⁽³⁾。本研究の主題である自動車用部品製造を始めとする機械的強度および耐磨耗性を要求される機械部品類の製造に多用される冷温間鍛造工程用の鍛造シミュレータの実用化段階においても、処理時間の大幅な短縮化が重要課題になっている。

最近、その解決策として並列処理技術が注目され始め、特に大規模な数値シミュレーションに要する計算時間を短縮化する有力手段として急速に期待が高まっている⁽⁴⁾。この並列処理の実行環境として、多数の汎用ワークステーション(以下、WSと略記)や高性能パソコンをネットワーク結合し、1組の分散メモリ型並列計算機として使用するワークステーションクラスタ(以下、WCLと略記)が注目されている⁽⁵⁾。

本報では、このWCLを用いた並列処理システムによって、大規模な鍛造シミュレーションで計算時間の大半を占める剛性方程式の求解処理⁽⁶⁾の大幅な高速化を目指している。一般に、WCLを用いた並列処理システムによる高速化の研究では、WCLを構成する各WS間のデータ通信にソケットが使用されるため⁽⁷⁾、個々の応用プログラム毎にソフトウェア開発をする必要があり、大規模システムになるほど多大な開発工数を要求されるのが現状である。

本報では、WCLを構成する各WS間のデータ通信の実現手段として、プログラミングが容易で、多様なシステムに柔軟に対応できるという特色を有するPVM(Parallel Virtual Machine)⁽⁸⁾と呼ばれる並列処理ソフトウェアを

* 原稿受付 1998年7月17日。

^{*1} 正員、岡山県立大学情報工学部(☎719-1197 総社市窪木111)。^{*2} 正員、九州工業大学工学部(☎804-8550 北九州市戸畑区仙水町1-1)。^{*3} 岡山県立大学大学院。

E-mail: matsuda@cse.oka-pu.ac.jp

採用⁽⁹⁾した。そして、本報で開発した鍛造シミュレータのための剛性方程式の並列求解法をこのシステム上に実装し、システム構成条件や通信方式を変更した場合の高速化効果について、多数の数値シミュレーション実験によって調べた結果を報告する。

2. 剛性方程式の求解方法

2-1 剛性方程式の規模について 本報では、図1の3次元塑性変形解析モデルに基づく剛性方程式の求解を主対象にしている。このモデルでは、解析対象の対称性を考慮に入れた1/8領域のみに注目し、同図(a)に示す角柱ハッチング部への押込み工具による圧縮加工を想定している。同図(b)は、加工途中(押込み率30%)の変形状態を例示したものであり、ここでは各辺を11等分している。この要素分割条件によって総節点数は $12 \times 12 \times 12 (= 1728)$ となり、自由度の総数は各節点の自由度が3になるため5184になる。なお、節点番号の順序は通例に従う自然な番号付けを採用した。この場合の剛性方程式の係数行列は正値対称帯行列になり、その最大帯幅は947である。

2-2 共役勾配法について 本報で並列高速化の対象とする剛塑性有限要素解析法を用いた大規模数値シミュレーションでは、剛性方程式に相当する連立一次方程式部分の求解時間の短縮化が最も重要である。そこで、この求解方法として、所要メモリサイズの点から有利で、分散型並列処理に多用される反復解法を採用し、その中でも最も代表的な共役勾配法(CG法)⁽¹⁰⁾を選択した。

共役勾配法に基づいて連立方程式 $Ax = b$ を解く場合の共役勾配法のアルゴリズムは、以下の第(1)~(10)項のような手順になる。ただし、記号 A , x_k , b , r_k , p_k は、それぞれ係数行列、近似解ベクトル、定数項ベクトル、残差ベクトル、修正方向ベクトルを表す。

- (1) 初期値 x_0 を選ぶ → (2) $r_0 = b - Ax_0$
 → (3) $p_0 = r_0$, $k = 0$ → (4) $\alpha_k = \frac{(r_k, p_k)}{(p_k, Ap_k)}$
 → (5) $x_{k+1} = x_k + \alpha_k p_k$ → (6) $r_{k+1} = r_k - \alpha_k Ap_k$
 → (7) $\beta_k = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}$ → (8) $p_{k+1} = r_{k+1} + \beta_k p_k$
 → (9) $k = k + 1$ → (10) 未収束 → (4) に戻る

このような共役勾配法では、上記アルゴリズムの第(4)~(9)項までの反復回数が全体の処理時間の大半を支配すると考えられる。剛塑性有限要素法においては、剛性方程式の係数行列である剛性マトリックスの条件数が弾性有限要素法などに比べて大きく、収束までの反復回数が増大する。そして、大規模解析ほど剛性方程式が高次数になり、求解に要する反復回数が急増してこ

Fig.1 Three dimensional deformation analysis for 11x11x11 elements: (a) before deformation, (b) during deformation.

れらが計算時間の飛躍的な増加につながる。

本報では、並列処理の適用によって反復計算の所要時間を短縮化して計算時間全体の短縮化を図る。ただし、共役勾配法では収束解に至るまでの各反復計算が一定計算量になるため、本報ではこの反復計算1回分の処理時間に注目して計算結果を評価する。

3. 並列化ツールPVMを用いたWCL

3-1 並列化ツールPVM 並列化ツールPVMは、統一化された枠組の中で既存の計算機資源を利用して並列処理を実現するための諸機能を提供するソフトウェアである。また、PVMはネットワーク接続された同一機種あるいは異機種から成るWS群を1組の仮想的な並列計算機、すなわちバーチャルマシンとして取り扱えるようにするものである。

このPVMを導入してWCLを構成・構築できれば、WCL全体を1組のバーチャルマシンと見なして操作できるため、PVMが提供する高度に抽象化された関数(サブルーチン)群を利用できようになる。このため、WCLの構成・構築作業を始めとして、WCL用の各種プログラム開発がソケットを使用する場合に比べ圧倒的に簡便かつ容易になる。こうしてPVM化されたWCLでは、その全構成WSの個々にpvmdと呼ばれる常駐プログラムを起動させておく。そして、各WS上のpvmdが相互に情報交換を行い、各WS間のデータ通信の仲介を行うことになる。このpvmdは、バーチャルマシン内で並列動作するプログラム群、すなわちPVMタスクに対する管理機能を備えており、pvmdに指示を与えることによりバーチャルマシン内の任意のWS上でPVMタスクの生成と消去が可能になる。このため、処理の規模、使用環境等の諸条件に応じてシステム構成を動的に変更し得るというような極めて柔軟性を持ったWCLを構築できる。

3-2 均質型のWCL 前述したように、本研究ではネットワークで相互接続した性能の等価な多数のWS

Fig.2 System layout of workstation cluster.

を、ソフトウェアによって1組の並列処理システムとして稼働させる分散メモリ型の並列計算機システム、すなわちWCLと呼ばれる計算機システムを使用している。このWCLを構成する各WS間ではネットワーク経由によるデータ交換を行うため、メッセージパッシングと呼ばれる方法で並列処理が実現される。

一般に、超並列計算機やWCLなどの分散メモリ型並列計算機は、共有メモリ型並列計算機に比べてプロセッサ数およびメモリ総容量の拡張が容易であると言われている。特にWCLは、システム全体のCPU数およびメモリ容量の拡張が容易であるだけでなく、汎用ネットワークを利用して結合されるため、高速ネットワークの増強・増設などによる機能強化や接続形態の変更が容易かつ柔軟に行えるという特色を持つ。このため、WCLは、これと対比される超並列計算機、すなわち1台の筐体内に多数のプロセッサを搭載し、それらの間のデータ交換に独自の高速通信機構を組み合わせる構成された市販の専用並列計算機などの高額システム類に比べると、はるかに実現容易な並列処理システムと考えられる。

図2は、本研究で使用するシステム構成の概要を模式的に示したものである。本図に示されたように、このWCLでは、各WSは互いに対等な関係で単一ネットワークに接続され、処理プログラムによる役割に応じてマスターとスレーブに区別される。なお、本研究では、WCLを構成する具体的なWSとしては、1台のWS内に4CPUが密結合方式で組み込まれた日本電算機(株)製のSpareStation20/model-514互換機を使用した。そして、各WS間を相互接続するネットワークとしては光通信式のFDDI (Fiber Distributed Data Interface) と等価なTPDDI (Twisted Pair DDI) を採用した。

3.3 共役勾配法の並列化 本報では、前節に述べたWCL上における共役勾配法の計算にマスター・スレーブ型の並列処理を適用することにより高速化することを試みている。共役勾配法による計算プロセスで

Fig.3 Parallelization of conjugate gradient method by workstation cluster.

は、係数行列 A と修正方向ベクトル p_k の内積部分が最も計算時間を要し、並列化による高速化効果が高まると考えられるため、この部分を複数のスレーブによる並列処理によって高速化する。

図3は、この処理の流れを模式図で示したものである。図示の通り、係数行列 A を総スレーブ数 N (以下、単に N と略記)で行方向に等分割し、各スレーブに割り当てられる行数が均一になるように分配する。各スレーブは、分配された部分係数行列と修正方向ベクトルの内積計算を行う。 N 個のスレーブは、分配された部分係数行列と修正方向ベクトルの内積結果をマスターに個別に送信する。マスターは各スレーブからの部分ベクトルを集計し、次ステップの修正方向ベクトル p_{k+1} を算出して各スレーブに送信する。

4. PVMにおける高速通信機能

4.1 PVMの標準通信機能 PVMは、PVMタスク間の標準通信機能としてpvmdと呼ばれる常駐プログラム経由による1対1の通信機能、すなわちユニキャスト通信機能を提供している。つぎの図4は、PVMタスク間のデータ交換の様子を模式的に表したものである。本図に示されたように、PVMタスクはpvmdに送信を委託し、pvmd間でデータの授受を行う。そして、相手先のPVMタスクはpvmdからデータを受け取る。

図5は、ソケットを用いたユニキャスト通信の場合とpvmd経由のユニキャスト通信の場合との処理時間の比較結果である。本図から明らかなように、ソケットの場合に比べると、pvmd経由の場合の方が全般的に処理時間が長くなり、 $N > 8$ の領域になると、このままでは実用性を欠くような結果となる。これは、pvmdを経由し

Fig.4 Data communication among PVM tasks in workstation cluster.

Fig.5 Execution time through *uni-casting* of pvmd and *uni-casting* of socket.

て通信が行われる場合のオーバーヘッドがソケットを用いる場合のそれに比べて増大するためと考えられる。この場合の実測値によれば、ソケットを用いる場合の方が、最大で53.4% (N=16)、最小で6.3% (N=1) だけ処理時間が短い。

しかし、最近のPVMには、通信時間の短縮に有効と思われるマルチキャストおよびダイレクトリンク(最近では、ダイレクトルーティングとも)と呼ばれる有用な機能が備えられているので、これらの適用を以下で検討する。まず、これらの機能の概要を以下の第4.2節および第4.3節に示し、両機能を適用した場合の高速化効果については第5章に論述する。

4.2 マルチキャスト機能 PVMでは、ユニキャストと呼ばれる1対1の通信以外に、1つのPVMタスクから複数のPVMタスクに同一データを単一命令で送信するマルチキャストと呼ばれる複合的な機能が提供されている。マルチキャストを用いた通信では、図6に

Fig.6 Data communication among PVM tasks through multi-casting.

模式的に示したように、複数のPVMタスクに対して同時に同一データを送信するのではなく、最小被覆木(Minimum Spanning Tree) アルゴリズムと呼ばれる方法に基づき、対象となるPVMタスクにpvmd経由で順次データを転送していく。

このマルチキャスト機能の適用により、下記の2条件(1)、(2)の一方でも成立すれば通信時間の短縮が期待できる。

- (1) 通信路のデータ転送能力が高くて複数組のPVMタスク間の通信をほぼ同時に実行可能な場合。
- (2) 図6に例示しているように、PVMタスク間のデータ転送に関し、物理的に複数組の異なる通信路を有することで同時通信が実行可能な場合。

なお、本報では、マスターから多数のスレーブへの同一データ p_k の送信過程が発生するため、ここにマルチキャスト機能を適用している。

4.3 ダイレクトリンク機能 PVMでは、ユニキャスト通信の実行手段として、pvmd経由の標準通信手順以外に、PVMタスク間で直接通信を行うダイレクトリンクと呼ばれる機能も提供されている。この機能を利用すると、PVMタスク間の通信がソケットを使って実行されるため、pvmd経由の通信に比べ大幅な高速通信が可能であると言われている⁽¹¹⁾。ただし、ソケットを使って通信する場合には、使用するネットワークおよび使用するWSのOS(オペレーティングシステム)によって、1回の通信で送受信できる最大データサイズが決められている。そのため、稼働中の各WSが許容するその最大サイズを逐一調べ、これを超える長大データについては、事前にユーザプログラム内でデータ分割して送信する必要がある。しかし、PVMを使用する場合には、それらの処理はすべてPVM側が自動的に行うため、プログラム作成上の負担を軽減できる。

5. 高速通信機能の効果

PVMを使用して並列化されたプログラムを稼働中のWCL上で実行させるのは、WCLを構成する各WS上で実行可能なPVMタスクと呼ばれるプログラムの協調動作によって成される。本報では、鍛造シミュレータ中で最も計算負荷の重い部分と考えられる剛性方程式の求解に用いる共役勾配法の内積計算部分をスレーブWS上のPVMタスクに分担させて並列処理化する。

本章では、本報の計算条件において使用可能なPVM機能の範囲内で、特に高速化に有効と考えられるマルチキャスト機能とダイレクトリンク機能を用いた場合の効果を調べる。以下で使用する処理時間の定義としては、共役勾配法の反復計算1回分に要する時間を表すものとする。処理時間の実測には、先の第2.1節に示した係数行列データを採用し、各通信方式毎にN数をN=1~16まで変更した場合の処理時間を調べた。

なお、本研究では、WCLを構成する各WSには4CPUを密結合したマルチプロセッサ型を採用している。このため、各スレーブWSに割り付けるPVMタスク数については、1~3個という条件に分けた各場合について実測を行い、各WSにどのようにPVMタスクを割り振るかというハードウェア構成条件の任意性についても注意を払った。

5.1 マルチキャスト通信による高速化効果 図7は、マルチキャスト通信を用いて同一WS内のPVMタスク数を1~3に変化させた場合の処理時間とpvmd経由のユニキャスト通信を用いた場合の処理時間とを比較した結果である。本図からわかるように、マルチキャストを用いた場合、 $N > 6$ の領域では、同一WS内のPVMタスク数が2および3の場合の処理時間がpvmd経由のユニキャスト通信を用いた場合より短くなる。ただし、 $N=2 \sim 4$ の領域では、同一WS内のPVMタスク数が少ない方がやや処理時間が短い。

このように、N数がある程度以上の領域では、同一WS内でのPVMタスク数が多い方が処理時間が短いという特性が得られる。このことについては、同一WS内のPVMタスク間の通信速度の方が、各WS間をまたぐPVMタスク間のそれよりも本質的に高速なために生じたものと推疑され易い。しかし、よく検討してみると、この特性は、以下の理由(1)、(2)によるものと考えられる。

- (1) 同一WS内の2つのPVMタスク間の通信は、ネットワークを経由しない。これにより、 p_k の送信処理において、ネットワーク経由の通信と同一WS内の通信を並列実行できるため、

Fig.7 Comparison of execution time through uni-casting of pvmd and multi-casting.

Fig.8 Comparison of execution time through uni-casting of pvmd and direct-linking.

- (2) 上項(1)により、単一ネットワークを介して同時に実行される通信の組数が減少したため、実効通信速度が高レベルに維持されるため。

5.2 ダイレクトリンクによる高速化効果 図8は、pvmd経由のユニキャスト通信およびダイレクトリンクによるユニキャスト通信を用いた場合の処理時間を同一WS内のPVMタスク数を1~3に変化させて調べた計算結果である。本図によると、処理時間は、両者ともに、同一WS内のPVMタスク数を1~3に増加させるに従い、僅かではあるが単調に増加するといえる。また、ダイレクトリンクを用いた場合の処理時間は、本計

Fig.9 Comparison of overhead time by using *uni-casting* of pvmd, *multi-casting*, *direct-linking* and *uni-casting* of socket.

算条件の範囲内では、pvmd 経由に比べ最大で約 1/2 に短縮されている。さらに、ダイレクトリンクの場合の最短処理時間は、マルチキャストの場合のそれ(図 7 参照)に比べて約 15% 短縮されている。

図 9 は、pvmd 経由、マルチキャスト、ダイレクトリンク、ソケットの 4 者を用いた各場合の処理時間から 1 スレーブ当たりの計算時間を除くことによって、全処理時間中の通信所要時間を含めたオーバーヘッド時間相当分を抽出した結果である。本図から、pvmd 経由、ダイレクトリンク、ソケットを用いた場合のオーバーヘッド時間は、N の増加に伴って漸増することがわかる。一方、ダイレクトリンクを用いた場合のオーバーヘッド時間は、pvmd 経由の場合に比べて約 60% 短くなり、全域にわたってソケットを用いた場合の結果とほぼ同等の値になる。また、マルチキャストの場合のオーバーヘッド時間は、ダイレクトリンクの場合に比べて全般的に大きくなっている。しかし、両者のオーバーヘッド時間の差は、N の増加に伴って漸減傾向で近づき、N=16 では僅少差となる。

つぎの図 10(a),(b) は、並列計算に要する全処理時間および並列処理化によって達成される計算の加速効果を調べたものである。特に同図(b)では、対象とする計算を 1-CPU 単独で行った場合の処理時間を基準にして、これに対する図中の各通信様式、すなわちソケット・ダイレクトリンク・マルチキャスト・pvmd の 4 者を組み合わせた 2 条件に対する個々の処理時間の比の値を計算加速率 α と定義して整理したものである。両図からわかるように、ダイレクトリンクによる結果は、N 数の

Fig.10 Comparison of execution time and speedup effect by using *uni-casting* of socket, *direct-linking*, *multi-casting* and *uni-casting* of pvmd : (a) elapsed time, (b) speedup.

全域でソケットの場合に近い値にまで接近する。この場合、後者が N=12 で最大加速率 $\alpha_{max} = 5.33$ になるのに対し、前者は、これより少ない N 数(N=8)で後者のそれに匹敵する $\alpha_{max} = 5.38$ を達成できたといえる。

以上のように、ダイレクトリンクと PVM とを併用した WCL では、従来のソケットを用いる場合の結果と同程度の高速化が達成可能であることが明らかである。

6. 計算結果に対する検討および考察

本報では、性能の等価な多数の WS を高速ネットワークで相互結合し、そこに並列化ツールである PVM を移植した均質型の WCL システムを利用可能にした。そして、剛塑性有限要素法に基づく大規模な鍛造数値シミュレーションの高速化に対する分散型並列処理システムの有効性を実測データに基づいて調べ、おおむね従来のソケット通信の場合に匹敵する良好な結果が得られ

たため、当初の目標は達成されたと考えられる。しかし、現有のシステム条件のままで、別報⁽¹²⁾に示されたようなデータ転送時の同時並行性を生かせる階層型モデルなどの新たな手法を導入すれば、さらなる計算加速率の増大につながる並列処理に特有の潜在能力を発揮できると思われる。

一方、原理的には高速化に大きな寄与があると期待されるマルチキャストを採用した場合の計算結果については、ダイレクトリンクを用いたユニキャスト通信の場合を上回るような高速化効果は確認できなかった。

しかし、マルチキャストを採用した場合のオーバーヘッド時間は、図9からわかるように、Nの増加に伴ってダイレクトリンクの場合とほぼ同等レベルに収まるような結果を得る。従って、マルチキャストは、本来ダイレクトリンクに比べかなり低速なpvmd経由の通信を利用しているにもかかわらず、総合的なデータ転送能力としては、上記のような好成績を達成できることになる。

このことから、PVMツール内でマルチキャスト用に準備された通信処理手順は、本研究で使用したWCLのような分散メモリ型並列計算機システムにおける計算実行時の通信所要時間の短縮化に非常に有効であると考えられる。そのため、PVMツールが今後機能強化され、マルチキャスト通信を高速なダイレクトリンクで実行可能なように発展を遂げれば、本研究のように単にダイレクトリンクをユニキャスト通信に使用する場合に比べ、高速化の達成率がより向上するものと思われる。

7. 結 言

本報では、ワークステーションクラスタ(WCL)に並列化プログラミングツールであるPVMを導入し、クラスタを構成する個々のワークステーション間のデータ通信方式に注目して、鍛造シミュレータの中核部分である剛性方程式の求解部の並列処理化を試みた。その結果、本報で検討した計算条件の範囲内では、ダイレクトリン

クを用いたユニキャスト通信の場合の処理時間が最短になることがわかった。また、通常のマスタ・スレーブ型モデルに基づくワークステーションクラスタに対してPVMのダイレクトリンク機能を適用すれば、ソケットによる従来法の場合と同等レベルの通信速度が達成できることが明らかになった。こうした研究成果によって、ソケットに比べると格段に取り扱いが容易なPVMを使用しても、比較的小規模なワークステーションクラスタにおいて高速処理化が実現可能なことを示せた。なお、本報で期待したマルチキャストを用いる場合については、ダイレクトリンクを用いたユニキャスト通信の場合を上回るような高速化効果は得られなかった。しかし、本文中で検討したように、技術的に有望と考えられるため、この件に関しては、さらに大規模計算領域を対象にして、今後早急に究明する必要があると思われる。

文 献

- (1) 王藤英明, 塑性と加工, **29**-324(1988), 4-12.
- (2) 寒川 光, 塑性と加工, **37**-421(1996), 140-145.
- (3) 湯川伸樹・岡田達夫, 塑性と加工, **37**-421(1996), 166-170.
- (4) 堀江知義・倉前宏行, 機論, **61**-584, A(1995), 861-868.
- (5) 矢川元基・内田善彦, 機論, **60**-580, A(1994), 2737-2742.
- (6) 宮川佳夫・松田 章・加藤 隆, 機論, **62**-603, A(1996), 68-73.
- (7) 堀江知義・倉前宏行, 機論, **62**-603, A(1996), 2462-2467.
- (8) Geist.G.A., ほか 5 名, PVM Parallel Virtual Machine, (1994), MIT Press.
- (9) 岩瀬英仁・神谷紀生, 機論, **64**-622, A(1998), 1589-1595.
- (10) 戸川 隼人, 共役勾配法, (1977), 1-16, 教育出版.
- (11) Geist.G.A., Parallel Virtual Machine -EuroPVM'96, (1996), 1-6, Springer.
- (12) 松田 章・加藤 隆・宮川佳夫・ほか 2 名, 機論, **64**-627, A(1998), 2701-2709.