32

超並列計算機T3Dのプログラム開発環境とCFDへの応用

木下　利博*　　斎藤　務*

Program Developing Environment and a CFD Application on the Cray T3D

by

Toshihiro KINOSHITA and Tsutomu SAITO
Cray Research Japan Ltd.

ABSTRACT

This paper addresses the performance evaluation of three-dimensional compressible inviscid flow analysis code on the Cray T3D massively parallel processing system. Second-order upwind TVD scheme and Strang-type time splitting method are employed in the analysis code. The method of data decomposition and array redistribution for minimizing the communication cost will be discussed. The outline of T3D system and its programming environments such as programming model and tools will be also given.

1. はじめに

単一プロセッサの実効性能に飛躍的な向上が期待できなくなっている昨今、科学技術計算の分野における並列計算機への期待は急速に高まっている。超並列という考え方そのものは決して新しいものではないが、そこに現在の新しい技術を投入した高性能の計算機が指向されてきている。

こうした超並列計算機（MPP）は、分散共有型のメモリの採用により、プロセッサ数に比例した（スケーラブルな）性能をできるだけ容易に得ることを目指して開発が行われているが、モンテカルロ・シミュレーションのように完全に並列な問題ばかりでなく、科学技術計算で最もよく使われる手法のひとつである差分法による計算においてもスケーラブルな性能が実現される必要がある。本報告では、差分法による解析の一例として、３次元非定常オイラー方程式をTVD差分法で解くコードを用いて、Cray T3Dにおける性能を評価した。

* 日本クレイ（株）

2. Cray T3Dシステム

2.1 T3Dシステムの概要

図1の概念図に示されるようにT3Dの各ノードは、３方向全てに関して３次元の格子の端を反対側の端に接続した、３次元トーラスと呼ばれる構造のネットワークによって接続されている。このネットワークは１方向当たり300MB/秒のバンド幅を持つため、それぞれのノードには６方向の合計1.8GB/秒のバンド幅が確保されてる。また、フロントエンドシステムであるCray Y-MPやCray C90へは、I/Oゲートウェイと呼ばれる専用のノードを介してチャネルで接続されている。このシステムのメモリは物理的には分散されているが、ハードウェアによる共有メモリ機構により、論理的には共有メモリシステムのように利用することが可能である。各PE(Processing Element)は、150MFLOPSの理論性能を持つDEC Alpha、16MBまたは64MBのローカル・メモリ、及び周辺回路から構成されており、最大構成の2048PEのシステムの理論性能は307GFLOPSである。

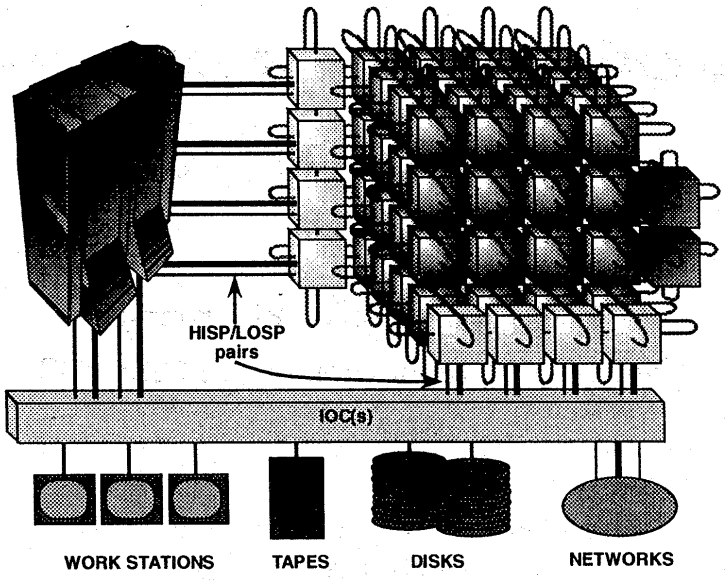


図1　Cray T3D システムの概念図

2.2 プログラミング・モデル

MPPの上でサポートされるプログラミング・モデルは、分散メモリ型と共有メモリ型の2つのモデルに大別される。分散メモリ型モデルは、初期のMPPから広く用いられてきたモデルであり、PE間の通信や同期を、明示的に記述したライブラリ・コールによって実現するスタイルのものである。従来のプログラムに対して付加的なプログラミングを多く必要とし、プログラミングはやや難しいと言える。これに対して共有メモリ型モデルは、従来の逐次処理システムに対するプログラミングにできるだけ近づけたもので、PE間の通信を明示的に記述する必要はなく、ディレクティブ（指示行）や特定の構文で、配列の分散方法や並列処理を指示するスタイルのものである。

Cray T3Dでは、分散メモリ型モデルとしては、並列計算システムにおいて標準となりつつあるPVM(Parallel Virtual Machine)を、また共有メモリ型モデルとしては、Fortran 90の配列構文に基づいたデータ・パラレルと、HPFやFORTRAN Dを参考にして設計された独自のディレクティブによって指示するワークシェアリングというプログラミング・モデルをサポートしている。さらに、共有メモリの機構を使って明示的に通信を記述するライブラリも用意されている。今回の解析コードの作成においては、この共有メモリ型のデータ通信ライブラリを利用してメッセージパッシングに近い形のプログラミングを行なった。

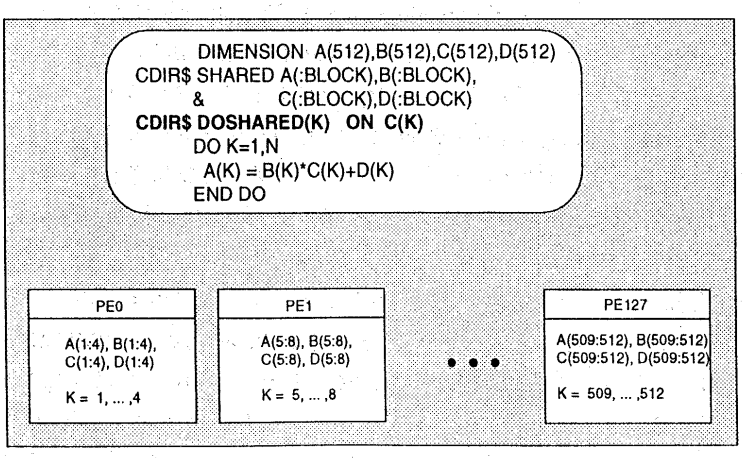


図2　ワーク・シェアリングによるプログラミング

2.3 プログラミング支援ツール

超並列計算機のプログラム開発においては、プログラミングそのものの難しさに加えて、並列計算機特有のバグの発見ということがしばしば問題となる。従って、並列計算機の特性にあった、会話型で使えるソースコード・レベルのシンボリック・デバッガが必要である。具体的には、PEごとにブレーク・ポイントの設定やステップ実行などの制御が可能で、変数の値などの情報がPEごとに容易に表示できることが不可欠である。また、性能解析に関しても、例えばPEが実際に演算を行っていた時間、あるいは他のPEの処理が終わるまで同期待ちしていた

などのオーバーヘッドの時間、キャッシュの利用状況などの情報が容易に把握できるツールが望まれる。Cray T3Dでは、並列計算機用のデバッガとして高い実績をもつBBN社のTotalView™、また性能解析ツールとしては独自に開発したMPP Apprenticeを提供している。

3. T3Dによる圧縮性流れの解析

3.1 解析コード

T3D用に作成した3次元非粘性圧縮性流れの解析コードは、一般曲線座標系上で定義された構造格子を用いて、3次元非定常オイラー方程式を陽解法で解くものである。解法には、Harten-Yeeによる2次精度TVDスキーム [1,2] を用いている。このコードでは3次元の問題に対して、Strang-typeの時間分割法 [3] を適用しており、時刻nにおける保存量ベクトルをQ^n、x, y, z方向の差分演算子をそれぞれL_x, L_y, L_zで表すとき、時刻n+2における保存量ベクトルQ^{n+2}は次式で表される。

$$Q^{n+2} = L_x L_y L_z L_z L_y L_x Q^n \qquad (1)$$

それぞれの演算子は1次元的な差分作用素であり、図3に示されているように、x方向の計算をする際には、矢印の方向には順番に計算しなければならないが、異なる矢印、即ちy、zが異なる格子点列あるいは領域は、並列に計算が行える。y方向の計算、z方向の計算も同様であり、領域分割によって容易に並列処理を実現できることがわかる。また、PE間のロード・バランスに関しては、各PEに割り当てられる格子点数が均等になるように領域を分割すれば、良好なロード・バランスを実現することができると考えられる。

3.2 領域分割の方法

x方向の計算を領域分割により並列処理する場合を考えると、図4のように3種類の分割方法が考えられる。1次元的な分割や2次元的な分割では、逐次処理を行うx方向のデータを、同一のプロセッサに持たせることが可能である。しかし、3次元的な分割では、計算（スイープ）を分断するような分割面ができてしまい、スイープの途中で同期をとって、通信を行う必要が生じてくる。また、スイープの後半部分の領域は前半部分の計算が終わってからでないと計算が始められないため、一部のPEがアイドル状態にならないように、領域へのPEの割り当て方には注意が必要である。1次元的な分割と2次元的な分割を比較した場合、1次元的な分割では、分割する方向の格子点数がPE数よりも多く、しかもPE数の整数倍になっていないと、PE間のロードバランスが極端に悪くなる場合があるのに対して、2次元的な分割では、PE数が多い場合でもロードバランスがとりやすいという利点がある。

3.3 配列の再分散

(1)式に示されるようなx,y,z,z,y,xという一連の計算の並列処理を考えると、x、y、z各方向の計算に適した領域分割の方法がそれぞれ異なるという点が問題になってくる。図4の1次元的な領域分割あるいは2次元的な領域分割を組み合わせて、計算の方向を切り替えるときに配列の再分散を行うか、または、どの方向の計算にも対応できるように3次元的な領域分割を行

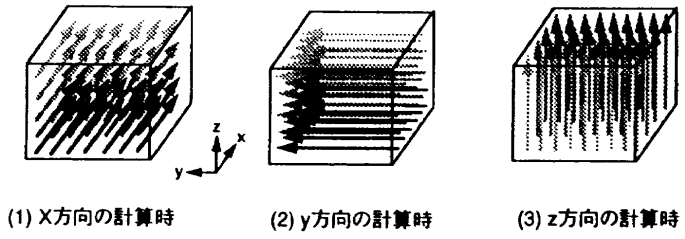


図3　解析コードの並列性

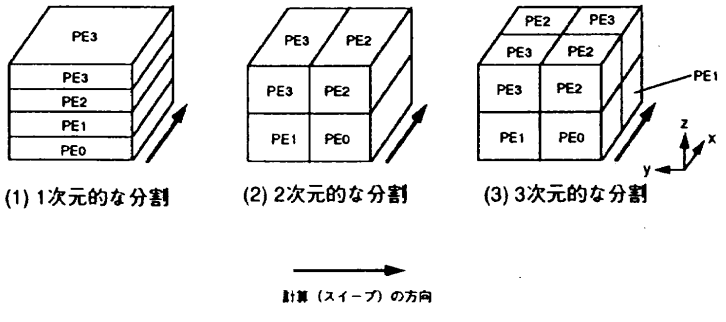


図4　x方向の計算を並列処理する際の領域分割の方法（4PEの場合）

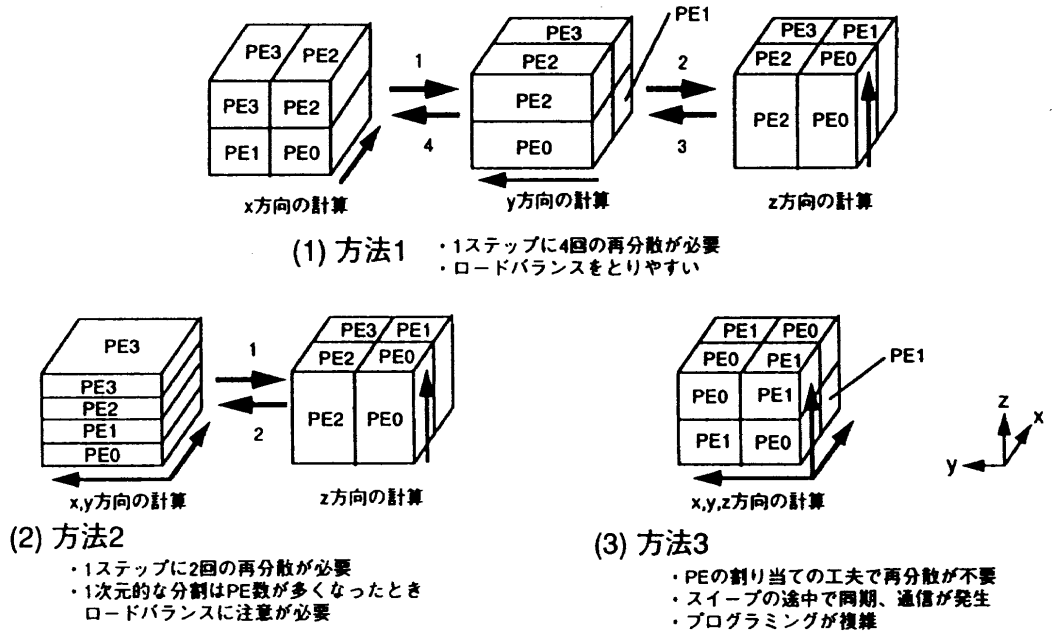


図5　配列の再分散

う必要がある。図5の方法1は、各方向に2次元的な領域分割を採用するもので、どの軸に関してもPE間のロードバランスをとりやすいという長所がある。しかし、図中に矢印で示されているように、1ステップに4回の配列の再分散が必要になる。方法2は、x方向の計算とy方向の計算に、同じ領域分割が使えるように1次元的な領域分割を採用したもので、1ステップ中の配列の再分散が2回で済むという利点がある半面、1次元的な分割はPE数が多くなったときにPE間のロードバランスが悪くなるという問題が残る。方法3は、グローバルな配列の再分散を行わずに、領域の境界部分のデータだけを転送することになるので、データの総転送量は少なくなる。しかし、逐次処理を行う場合と同一のスキームを使用するとすれば、スイープの途中で同期をとる回数がPE数に応じて増加していくことになる。さらに、どの方向の計算にも並列処理を行うためには、図6のように分割された領域のPEへの割り当てが複雑になり、プログラミングが困難になると同時に、PE数が多くなった場合に、やはりPE間のロードバランスの問題が生じてくる。

配列の再分散を行う際の通信パターンは非常に複雑で、ネットワーク上でのコンテンションも大きいと考えられるため、配列の再分散の回数は少なくする必要があり、解析コードの作成にあたっては図5の方法2に示される領域分割及び配列の再分散を採用することにした。

グローバルな配列の再分散が1ステップあたり2回

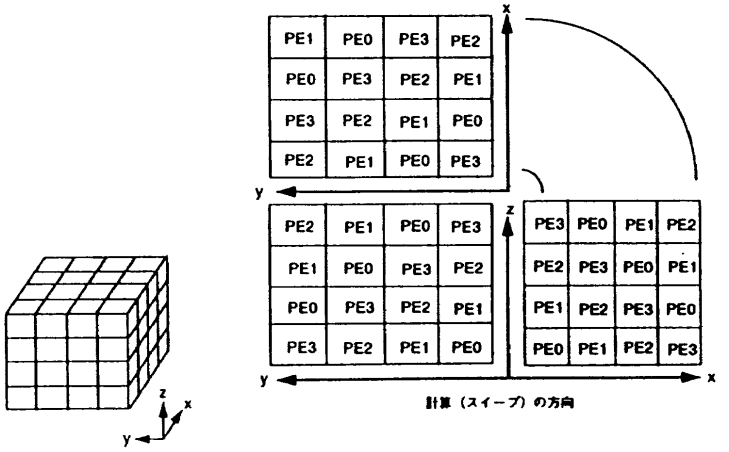


図6　方法3採用時の領域のPEへの割り当て例（4PEの場合）

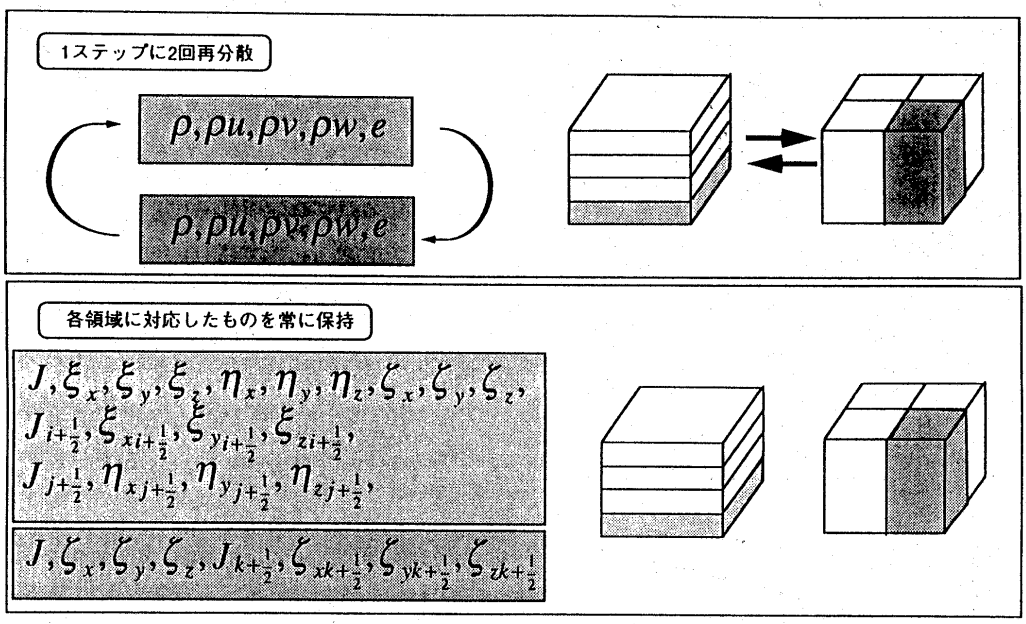


図7　データ通信量の低減

必要であるが、データ通信量を極力小さくするため、再分散を行うのは保存変数（密度、モーメンタム、エネルギー）に限定した。座標変換のメトリックスやヤコビアンは、流れ場の計算中に更新される量ではないので、図7に示されるように、2種類の領域に対応して必要なものを各PEが常に保持するようにした。座標変換のメトリックスなどを計算する際に必要な格子点の座標値も、同様な2種類の領域分割を行ったが、セルの界面でのメトリックスを計算する際には、他の領域に属する格子点の座標値も必要になってくる。このため、各領域に図8のようなゴーストセルを設けて、隣接格子点の座標も予め持たせるようにした。

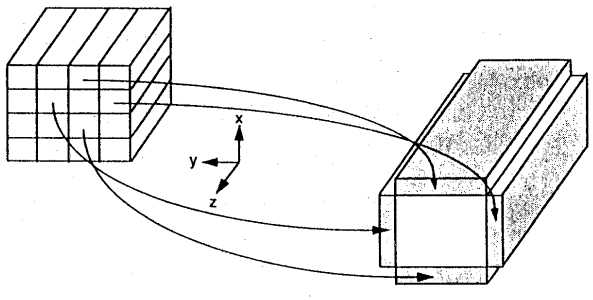


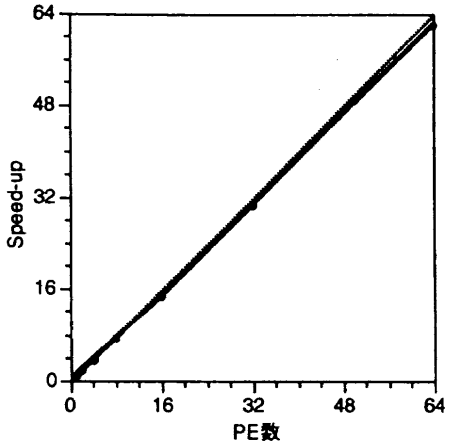
図8　格子点座標値の領域分割

4. Cray T3Dにおける性能

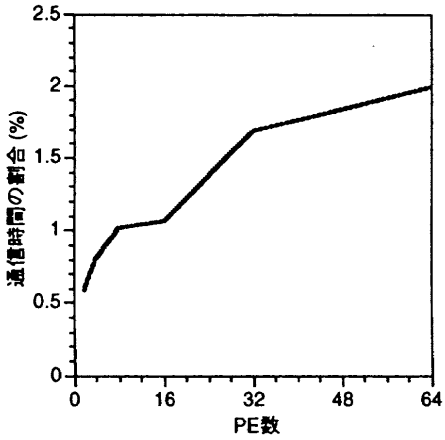
解析コードのT3Dにおける評価として、まず同一の問題を対象に、使用するPEの数を変えて性能を調べてみた。性能の評価に用いた問題は、マッハ数1.5の平面衝撃波が90°曲がり円管に入射した後の挙動の解析である。計算格子のトポロジーは直方体であり、格子点数は32×32×128の約13万点である。1ステップあたりにグローバルに転送されるデータの総量は約10.5Mbytesで、PE数を増やしていった場合、PEあたりのデータ転送量は線形に減少していく。

この計算例でのPE数とスピードアップの関係は、図9(a)に示されるように非常に線形に近いものとなった。同じ問題を解く際に使用するPEの数を増やしていくと、1つのPEに割り当てられる領域は小さくなっていくため、キャッシュのヒット率が向上し、1プロセッサあたりの性能は向上していく。一方、同期をとる際のオーバーヘッドなどは、PE数の増加とともに増大していくため、両者の効果が相殺して結果的に線形に近いスピードアップが得られていると考えられる。PEあたりの問題の規模が小さくなると、通信時間の占める割合が増えていく傾向にあるが、1%から2%の程度であり、演算時間に比較して十分小さい。

次に同様の問題を、PEあたりの格子点数が一定になるように、PE数の増加に応じて問題の規模も増大させるという条件で評価を行った。PEあたりの格子点数は

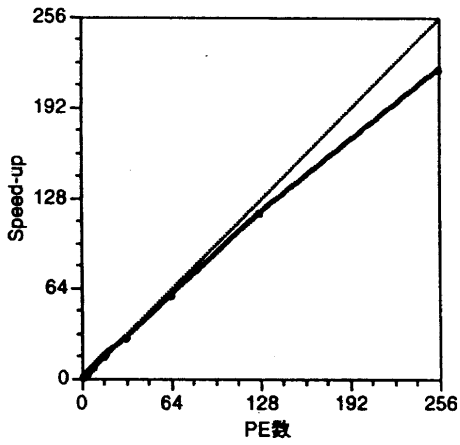


(a) PE数とスピードアップ

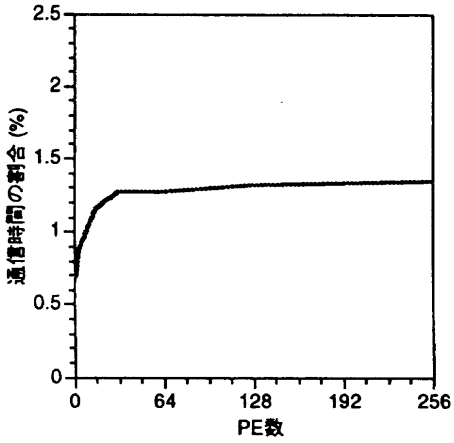


(b)通信時間の割合

図9 問題のサイズを固定した場合の性能



(a) PE数とスピードアップ



(b)通信時間の割合

図10 問題のサイズを線形に変化させた場合の性能

約16,000点とし、128PEのとき約210万点、256PEのとき約420万点の問題を解いた。PE数に応じて解く問題が異なるので、格子点一点あたりの計算時間で性能の比較を行った。

この場合のPE数に対するスピードアップは、図10(a)に示されるように、PE数の増加とともに線形な値から遠ざかっているものの、256PEという比較的多数のPEを使った場合でも、スピードアップのカーブは鈍化していない。各PEに割り当てられる領域の大きさ、及びPEあたりのデータ転送量が一定であるため、通信時間が占める割合もほぼ一定であり、約1.3%という低いレベルにとどまっている。

5. まとめ

一般に通信はMPPのボトルネックになりやすいが、今回作成した3次元非粘性圧縮性流れ解析コードでは、通信に要する時間は演算時間に比較して十分低く抑えられていることが確認できた。従って、解適合格子を採用するなど、MPP上では動的な負荷分散が必要となるコードへの応用も期待できる。

参考文献

1) Yee, H.C., *Upwind and Symmetric Shock-Capturing Schemes*, NASA Technical Memorandum 89464.

2) Harten, A., *High Resolution Schemes for Hyperbolic Conservation Laws*, Journal of Computaional Physics 49, (1983), pp.357-393.

3) Strang, G., *On the Construction and Comparison of Difference Schemes*, SIAM J. Numer. Anal. Vol. 5, No. 3, September 1968.