DNSコードの並列ベクトル化とその性能評価

山崎裕之*1、吉田正廣*1、岩宮敏幸*1
Grégory Coussement*2, Bruno Troff*2, Juliette Ryan*2, Paul Sagaut*2

Parallelization and Vectorization of DNS Code and Performance Evaluation

by
Hiroyuki Yamazaki*1, Masahiro Yoshida*1, Toshiyuki Iwamiya*1,
Grégory Coussement*2, Bruno Troff*2, Juliette Ryan*2, Paul Sagaut*2

ABSTRACT

We parallelized a DNS solver based on the Navier-Stokes equations in two ways: one uses PVM message passing library and the other NWT-Fortran compiler directives. Employing both 1- and 2-dimensional partitioning methods we analyzed the performance of the program on NAL vector parallel computer "NWT", and applied vector tuning and parallel tuning to it to be suitable for NWT. Then we made a performance model of the program to estimate the consuming time of the program. Comparing the estimation time derived from the model and the actual time consumed, we evaluated applicability of the performance model and obtained a good agreement. This work was done in the collaboration between ONERA and NAL. The program treated in this work is called "PEGASE" developed by ONERA.

1.はじめに

本研究の目的は、ベクトル性能、並列性能の評価のためのモデルを作り、これに基づいて処理時間の推定値を求め、実測値と比較、検討することである。本研究は航技研とフランスの航空宇宙研究所であるONERA との共同研究の一環として行われており、ONERAが開発したPEGASEという乱流の直接シミュレーションプログラムを対象としている。ベクトル処理性能、並列処理性能を推定するためにPEGASEの処理時間を推定する評価モデルを作った。また、航技研に設置されている並列ベクトル計算機「数値風洞(NWT)」上での実効性能を向上させるためにプログラムの改造、改良を行った。これらから推定性能と実測性能の比較、検討を行った。

2.方程式、境界条件、計算領域

「PEGASE」は、非定常非圧縮3次元ナビエ・ストークス方程式

$$\nabla\cdot\vec{v}=0$$

$$\frac{\partial\vec{v}}{\partial t}+\nabla(\vec{v}\otimes\vec{v})=-\nabla p+\frac{1}{\mathrm{Re}}\nabla^2\vec{v}$$

に基づき乱流の直接シミュレーションを行うプログラムで、

$$-\nabla^2 p=\nabla\left(\nabla(\vec{v}\otimes\vec{v})+\frac{\partial\vec{v}}{\partial t}-\frac{1}{\mathrm{Re}}\nabla^2\vec{v}\right)$$

$$\frac{\partial\vec{v}}{\partial t}+\nabla(\vec{v}\otimes\vec{v})=-\nabla p+\frac{1}{\mathrm{Re}}\nabla^2\vec{v}$$

と変形してpressure poisson 解法の形に直して解くものである。 図1にプログラム処理の流れを示す。

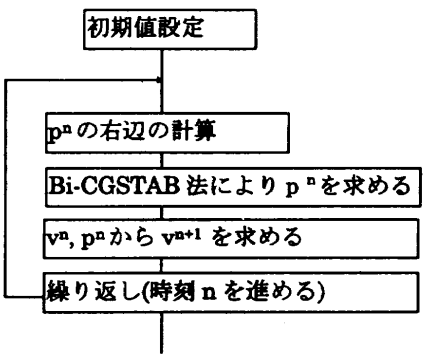

図 1 原PEGASEプログラムの動作

まず、初期速度場(v_0)を与える (初期圧力場(P_0)は0)。時刻nの速度場(v_n)から時刻nの圧力

*1 航空宇宙技術研究所
*2 ONERA(Office National d'Etudes et de Recherches Aérospatiales)

場（P_n）を求める。v_nとP_nから時刻n+1での速度場（v_{n+1}）を求める。圧力場を求める時の1次方程式の解法には「Bi-CGSTAB 法」を用いている。Bi-CGSTABの反復計算の中では、

- PE内の全点を対象とした内積計算（1時刻進める計算の中に5回）
- 各点毎に独立な演算（隣接点の値を参照しない）
- 2階差分から得られる1次方程式を解く（1時刻進める計算の中に2回）

の計算が含まれている。

今回評価の対象とした乱流場は2次元無限チャンネル流で、境界条件は上下の壁面で固体壁条件、その他は周期境界条件である。上下方向をz-軸に、残りをx,y-軸とする。プログラム上ではそれぞれ、K-方向（上下方向）、I,J-方向（その他の方向）とした。

計算領域は1個の直方体でこれを等分割した。処理性能の計測では計算領域の大きさを、全体の問題規模を固定して使用するプロセッサ（PE）台数の増加に伴う処理性能の速度向上を見る「スピードアップケース」では127×63×256、1PEが担当する問題規模を固定してPE台数の増加に伴う処理性能の加速性を見る「スケーラビリティケース」では1PEでの問題規模が70×42×85、となるようにした。

3.並列化

並列プログラムとしては、メッセージパッシングライブラリ（PVM）を用いて並列化したプログラムと数値風洞の並列処理記述言語であるNWT-FORTRANの並列化指示行を用いて並列化したプログラムの2種類を作成した。

図 2 計算領域の分割方法と境界条件

計算領域の分割法としては、NWT-FORTRANを用いた並列化ではK方向への1次元分割を、PVMを用いた並列化では同じ1次元分割とI、K 方向への2次元分割を用いた(図2)。並列化後のPEGASEの処理の流れを図3に示す。

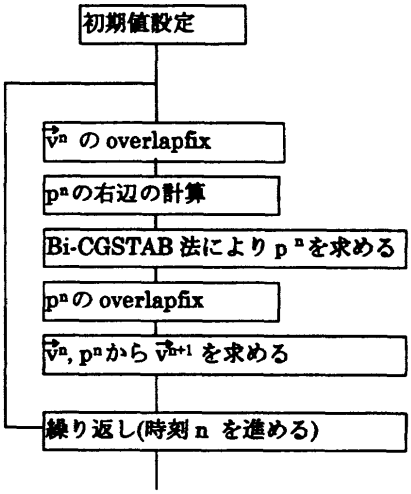

図 3 並列化後のPEGASEの処理の流れ

ここで、v_n、P_nの「overlapfix」は計算領域の分割に起因する隣接 PE 間での境界データの相互転送を表す（NWT-FORTRANではこれを「overlapfix」というコンパイラ指示行により実現する）。1次元分割の場合には、周期境界条件の方向には計算領域を分割しないので、これに起因するデータ転送は生じない。しかし、2次元分割では1つの周期境界方向も分割されているため、その方向にも隣接PE間での通信が必要となる(端を分担している2つのPEも計算領域の周期性から“隣接”していると理解するか、並列計算機のアーキテクチャによっては積極的に“隣接していない”と理解するか、は別の問題である)。Bi-CGSTAB計算の中でも、2階差分から得られる1次方程式を解く際に、1時刻進行の中で2回のoverlapfixに起因するデータ転送が生じる。この他にBi-CGSTAB 計算の中で内積計算に関連したグローバル処理に依るデータ通信が発生する。

4.ベクトルおよび並列チューニング

今回対象としたPEGASEプログラムは、もともとベクトル計算機に適したコードではなかったため、NWTでの性能評価をするためにベクトルおよび並列チューニングを行った。

4.1 ベクトルチューニング

ベクトルチューニングの例として、NWTのハードウェアが持っている「マスク演算処理機構」による例を示す。2次元分割法のプログラムはPE台数をK-方向、I-方向と交互に増やすようになっている。このためスピードアップケースでは、I方向にベクトル処理するDOループのベクトル長がPE台数の増加と共に短くなり（表1）、ベクトル処理性能が低下する。そこでマスク演算機構を利用して、引き続くIとJのDOループを1つのループとしてコンパイラが認識できるようにすればベクトル長を長くすることができ、ベクトル処理性能が改善される（図4）。

表 1 マスクベクトル化採用の有無によるベクトル長の相違　全体の大きさ 70 × 42 × 85 (*:17または18)

マスク化	1PE	2PE	4PE	8PE	16PE
なし	70	70	35	35	17/18*
あり	2940	2940	1470	1470	714/756

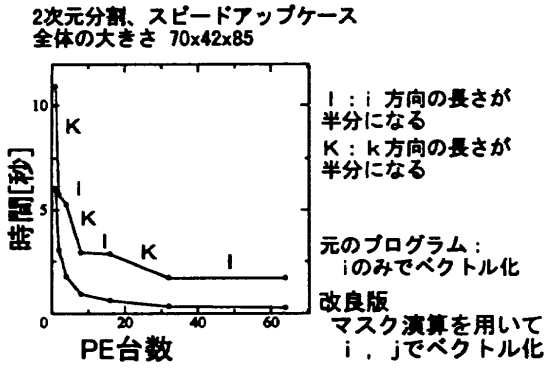

図 4 マスクベクトル化による性能向上

4.2 並列チューニング

次に並列チューニングの例として、内積計算を示す。内積計算の並列化では、それぞれのPEが自身の内部にあるデータに関する内積計算をした後、それぞれのPEで得られた結果を集めて足しあわせるというグローバル処理が必要となる。原プログラムでは、それぞれのPEが自分の持っている結果を順次隣におくるという動作で全体和を求めているため、PEの台数に比例した時間がかかっていた。これを2進木方法に改めた（図5）。これにより、台数が2の巾のときは $\log_2$PE の台数分の時間で済むことになる（図6）。

原プログラムと並列ベクトルチューニング版との結果を図7、8に示す。

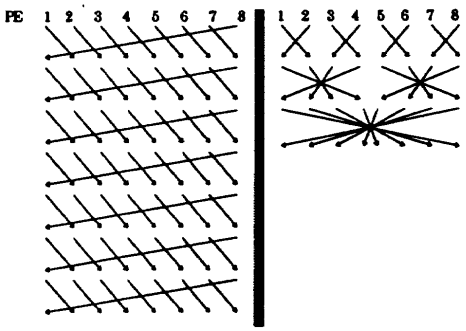

図 5 通信方法の改善
(左:元のプログラム、右：2進木方法)

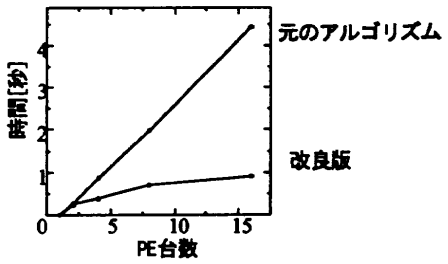

図 6 グローバル通信時間の改善（一回あたり）

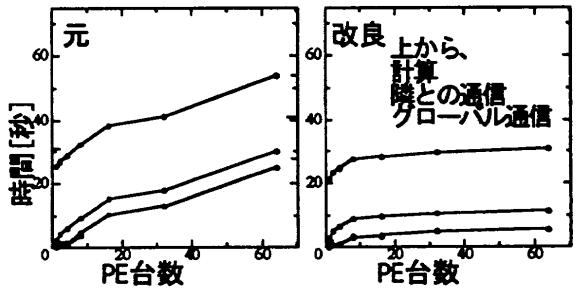

図 7 プログラム全体の性能向上 (PVM、2次元分割、スケーラビリティケース、1PEの大きさ 70x42x85)

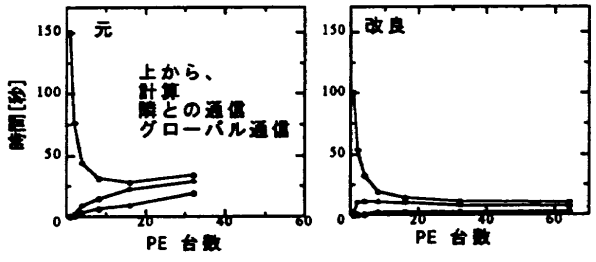

図 8 プログラム全体の性能向上 (PVM、2次元分割、スピードアップケース、全体の大きさ 127 × 63 × 256)

5.性能評価

プログラムの性能評価のために、プログラムを計算時間の大部分をしめるベクトル処理部分と通信部分に分ける。それぞれ別個に所要時間の評価モデルを作り、そのモデルに基づいて所要時間を推定し実測時間と比較検討する。

5.1 ベクトル性能評価

```
DO i=1,N
  S(i)=R(i)-(ml*P(i))
ENDDO
```

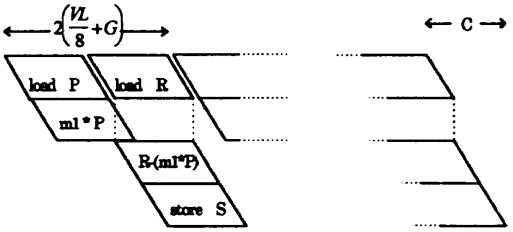

図 9 ベクトル計算とその動作

ベクトル処理性能の評価の例として図 9 のループを考える。この例では、Pをベクトルレジスタにロードし、Pと定数 m_1 の乗算を行い、この間にRをロードし（大抵のベクトル計算機ではロード／ストアと乗算、加減算が並列処理可能）、Rと乗算結果の減算を行い、演算結果をメモリにストアする。NWTの場合、その推定処理時間 T_E は第1次近似では以下の式で表される。

$$T_E=\left[2\times\left(\left(\frac{VL}{8}+G\right)\times\left\lfloor\frac{N}{VL}\right\rfloor+\frac{\mathrm{mod}(N,VL)}{8}\right)+C\right]\times\tau$$

ここで、乗数の2はベクトル命令の数、VLはベクトルレジスタ長、乗数8は1つの演算パイプラインの多重度、Gはベクトル命令発行間隔のクロック数、Nは実ベクトル長、C は立ち上がり／立ち下がり、τ はクロックサイクル時間である。ここでG以外はプログラムで与えることが可能な量なので、これらを変化させて実測することによりGの値の推定ができる。得られた結果は G=2.7 クロックである。 Bi-CGSTABの計算部分は3つに分けられるが、ここで得られたGの値を用いて他の計算部分についても推定した結果を表2、表3に示す。計算領域を1次元分割した並列化プログラムでは、並列処理記述に用いた言語－NWT-Fortran と PVM－に関係せず、計算部分のコードは基本的に同じものであるため推定値は同じである。2階差分の処理部分の実測値が推定値よりも小さくなっているが、これはコンパイラの最適化でレジスタの再利用が行われているためと思われる。また並列化記述言語による違いもコンパイラに起因するものと思われる。

表 2 推定値と実測値の比較（NWT-FORTRAN、PVM 1次元、Np は PE 内の格子点数）

	推定値	NWT-Fortran	PVM 1次元
2階差分	14.85 Np	12.61 Np	13.17 Np
各点ごとの計算	10.81 Np	10.95 Np	10.97 Np
内積	12.02 Np	12.66 Np	12.75 Np

表 3 推定値と実測値（PVM 2次元）

	推定値	PVM 2次元
2階差分	17.10 Np	19.90 Np
各点ごとの計算	10.81 Np	11.44 Np
内積	12.02 Np	13.28 Np

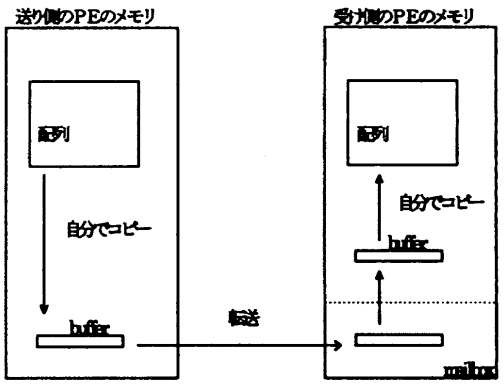

図 10 PVM ライブラリを用いた通信の内部動作

5.2 通信時間の評価

PVM を使った通信動作を図 10 に示す。1回のデータの送受信を行うためには、まず配列から送りたいデータを取り出し、1次元の作業配列に転送する。これは送信データが連続アドレスになっていない時のためである。1次元配列に転送したデータを SEND サブルーチンで送信する。データはまず相手のメモリのメールボックスと呼ばれる領域に格納さ

れる。受信側が RECEIVE サブルーチンを呼び出すとそのパラメータに指定された作業配列にコピーされる。そのあと、受信側は自分で必要な配列領域にデータをコピーする。これは送信側のときと同様、受信側の処理プログラムがデータを連続アドレスで処理しない時のためである。

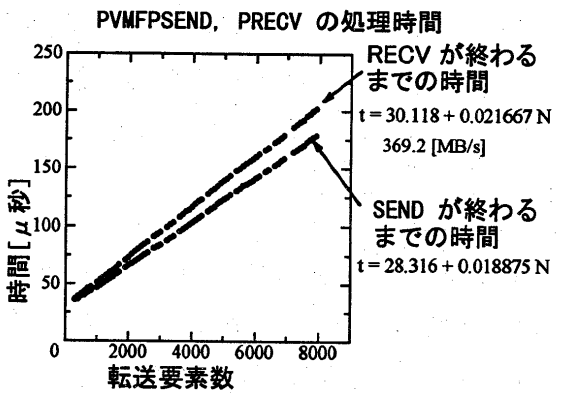

図 11 PVM 通信の要素数と通信時間

通信時間は、送受信 PE が行う送受信配列と作業配列間のコピーに要する時間、(レイテンシ+転送データ量に比例した転送時間)で表される PE 間でのデータ転送時間－SEND サブルーチンの実行時間－、受信側 PE が行うメールボックスと作業配列間のコピー－RECEIVE サブルーチンの実行時間－の和になる。NWT の場合、コピー時間はバンク競合、バス競合によりメモリを連続アドレスでアクセスするかどうかで処理性能が変わり、連続でないときは性能が落ちる。SEND、RECEIVE サブルーチンを使って、それぞれのサブルーチンの処理に要する時間を測定し、その近似式を得た（図 11）。2 次元分割した場合の overlapfix 操作を PVM の SEND、RECIEVE を使って行った場合の動作を図 12 に示す。上半分は I-方向へのデータ転送、下半分が K-方向へのデータ転送である。図 12 からわかるように I-方向のデータ転送では PE 内コピー時間が大きくなっている。この時には送受信されるデータが 3 次元配列の 2 次元目と 3 次元目で指定される境界面の要素となる。このため、ストライド付きのロード／ストアになるため処理性能が落ち、長いコピー時間となる。K-方向のデータ転送では、1 次元目と 2 次元目で指定される境界面の要素が転送されるため、連続アドレスデータのロード／ストアとなり、処理性能の低下は見られない。

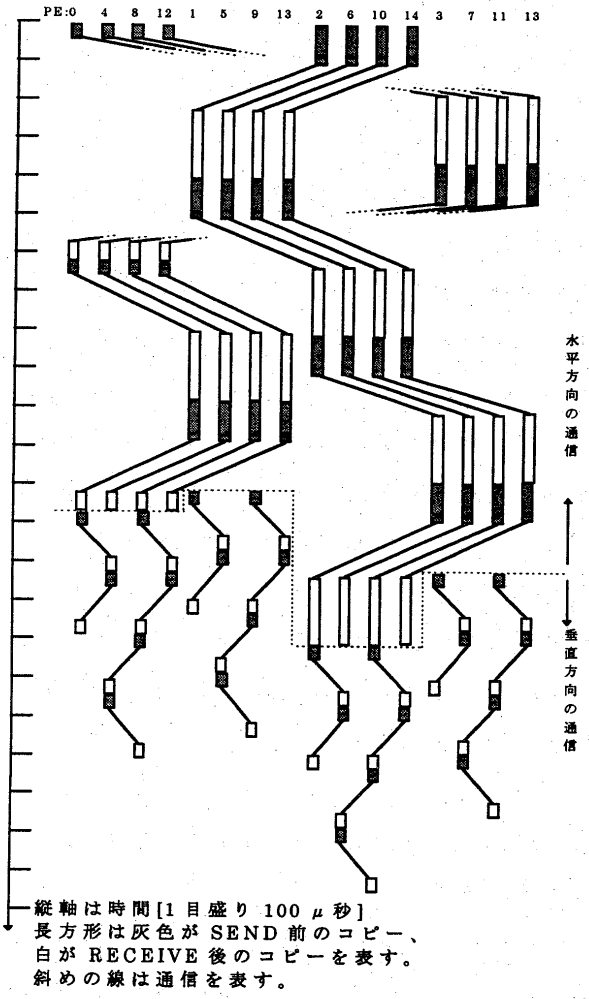

図 12 通信の動作（縦軸は時間[100 μ秒]、斜め線は通信、長方形は PE 内コピー時間）

このような通信動作を考慮して最も時間のかかるパスを持つ PE の通信時間の実測値と、モデルに基づく推定値とを比較したものが図 13 である。

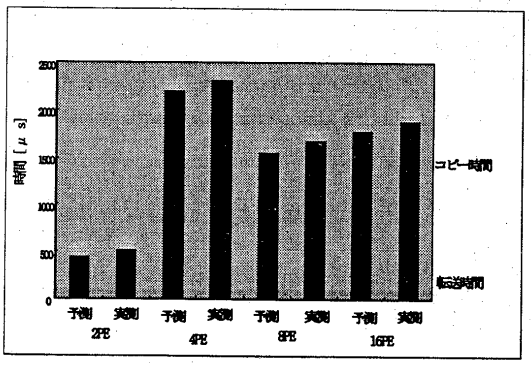

図 13 PVM 2 次元分割の 1 回あたりの通信時間（スピードアップケース 127 × 63 × 256）

この場合の転送時間の増減は次のように説明される。まず2 PEから4 PEへPEが増加すると転送時間も増えるのは、I-方向への転送が発生しコピー性能が大幅に落ちることと、転送相手、転送量が増えるためである。 8 PEへPEが増加すると転送時間が減るのは、I-方向の転送量が半分になって処理性能の悪いストライド付きコピーの所要時間および転送時間が減少するためである。 16PEの時にまた転送時間が増加するのは、K-方向の転送回数が増加するためである。また転送時間については比較的良く推定されているのに対し、コピー時間は差が大きくなっている。特に連続アドレスのコピーのときに推定時間に比べて実測時間が長くなっている。これはループ制御などのスカラー命令の実行等のオーバーヘッドが相対的に大きくなって今回のベクトル性能推定の範囲ではカバーされていないためと考えられる。図14、図15にプログラム全体に対する実行時間の推定値と実測値を示す。

表 4 PVM2次元のoverlapfix 1回あたりの通信の要素数と相手の数

	2PE	4PE	8PE	16PE
要素数	8001	12096	8064	6048
通信の相手の数	1	3	4	4

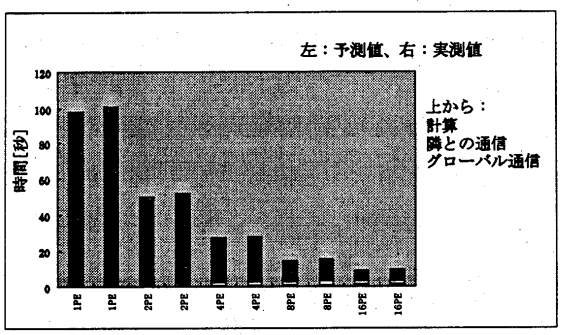

図 14 プログラム全体に対する推定時間と実測時間（スピードアップケース、PVM 1次元、全体の大きさ 127×63×256）

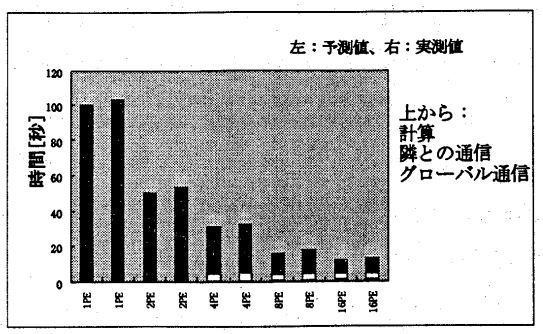

図 15 プログラム全体に対する推定時間と実測時間（スピードアップケース、PVM2 次元、全体の大きさ 127×63×256）

6.まとめと今後の課題

PEGASEプログラムに対して、プログラムの性能上の問題点を解析し、ベクトルチューニング、並列チューニングを行い、性能を向上させた。

ベクトル計算と通信にかかる時間のモデルを作り、実測値と推定値を比較しモデルの評価を行った。この結果ベクトル長が長い時には本モデルでの推定が有効である事が分かった。ベクトル長が短い時にはスカラ処理などベクトル演算にとってのオーバーヘッドをより良く推定することが必要である。通信時間については今回作ったモデルが有効である。今後はPEGASE以外のアプリケーションにおける計算パターンに対してもモデルを適用し推定対象の範囲を拡大すると共にその精度を高めていきたい。

参考文献

1. G.Coussement, B.Troff, J.Ryan, 山崎裕之、吉田正廣、岩宮敏幸 “High Performance Computing for Turbulence Simulation in Computational Fluid Dynamics”, HPC Asia ‘97
2. G.Coussement, B.Troff, J.Ryan, P.Sagaut, 山崎裕之、吉田正廣、岩宮敏幸 “DNS Code Implementation on High Performance Computers”, AFOSR International Conference, Louisiana Univ.
3. 中村孝、吉田正廣、山崎裕之 “行列積における並列処理性能の評価”、第14回航空機計算空気力学シンポジウム