# 並列システムにおける実効性能推定法について

松尾裕一[1]，末安直樹[2]，稲荷智英[2]，小林穣[2]
[1]宇宙航空研究開発機構，[2]富士通株式会社

# A Sustained Performance Prediction Method on a Parallel Computing System

by
Yuichi MATSUO, Naoki Sueyasu, Tomohide Inari and Minoru Kobayashi

**ABSTRACT**

In this paper, we first show the performance evaluation results on a large clustered SMP system consisting of the Fujitsu PRIMEPOWER HPC2500 installed to JAXA with 56 SMP compute nodes with 32 CPUs each for typical hybrid parallel CFD codes, and discuss the correlation between the code features and the parallel scalable performance. Next, we propose a sustained performance prediction method for the hybrid parallel programming by making the extension of Amdahl's law, and show the validity of the performance prediction by comparing the scalability performance data of the JAXA CFD codes and the prediction data from the proposed extended Amdahl's law. We found that the extended Amdahl's law is useful for the performances prediction for the JAXA CFD codes with less communication, and that the enhanced version of extended Amdahl's law with inter-node communication effects included is valid for the communication intensive codes.

## 1．はじめに

宇宙航空研究開発機構（以下，JAXA）では，旧航空宇宙研究所時代の1980年代後半から，スーパーコンピュータの比類ない計算能力を利用して，計算流体力学Computational Fluid Dynamics (CFD)に代表される数値シミュレーション技術を先駆的に研究開発し，流体基礎現象の解明や航空宇宙機の設計開発に適用してきた．1993年には，130台の要素計算機とピーク性能230GFLOPSを有する分散並列ベクトル計算機「数値風洞」を導入し，並列CFDの礎を築くとともに実問題への一層の適用を推進してきた．最近のCFDアプリケーションの傾向としては，工学系の解析では，設計への高度適用に係る現実の物体形状を見据えた複雑形状への対応が進んでおり，500万～数1,000万メッシュ上で，マルチブロック構造格子や非構造格子を用いることにより，エンジンナセルやフラップのついた機体まわりの流れ解析やエンジン内部の複数段の流れ解析が可能になって来ている．また，時間とともに現象が変化したり，物体が移動する非定常（過渡的）問題や，流体ー構造などの多分野連成問題が扱われるようになって来ており，いずれの場合も既にプロダクションレベルでの実設計開発への適用が行われている．一方，学術系の解析では，マルチスケール・マルチフィジクスといった現実の現象をできるだけ正確に取り扱う解析の方向へと進展しており，乱流や燃焼流のシミュレーションでは，1,000万～10億メッシュ上で高速流や化学反応を考慮しながら，出力データは量にして時に100GBを超えるようなケースも現出している．並列計算の観点からは，単純な幾何分割やデータ構造の分割から，計算領域を分割して各領域をCPUにマッピングする並列化手法やMPIによる転送が主流となりつつある．その一方で，通信負荷の重いFFTや補間処理も依然として使われている．

こうした状況の中で，2002年10月からは，富士通PRIMEPOWER HPC2500を中核とする全部で2,304個のCPU有する大規模SMPクラスタを導入した．このうち計算を担当する部分は，32個のCPUで構成するSMPノード56台から成り，9.3TFLOPSの計算処理性能を有する．このシステムでは，ノード内は共有メモリ並列（スレッド並列），ノード間は分散並列（プロセス並列）を組み合わせたハイブリッド並列を標準の並列化スタイルとして採用している．ハイブリッド並列では，ノード内のスレッド並列に関してコンパイラによる自動並列化機能を用いることができるなど，特にプログラミングの点で有利と言われている．確かに，ベクトルの前システムからスカラーの今システムへ移行する際，ベクトルループをスレッド並列で置き換えることにより混乱なく移行を実現できた．しかし，性能に関しては，ハイブリッド並列は，純MPIによる並列化を上回る性能は得られないという報告[1]もある一方で，JAXAの並列CFDコードでは，ハイブリッド並列の方が純MPI並列より性能が良いケースもあり（後述），計算資源の有効利用という観点からは，コードの特性パラメータと並列性能の相関を把握し，ハイブリッド並列に有効な性能モデルや簡便な性能推定法を見出すことが重要である．

本報告では，JAXAにおける並列CFDコードの性能測定結果を示すとともに，コードの特性と並列性能の関係について考察する．さらに，アムダールの法則を拡張したハイブリッド並列における簡易な実効性能推定法を提示し，その推定精度を検証し有効性を示す．

## 2．CeNSSの構成とプログラミングスタイル

JAXAに導入されているスーパーコンピュータシステム「数値シミュレータIII（NS-III）」の全体構成を図1に示す．このうち，計算部分を担当するサブシステムは，Central Numerical Simulation System（CeNSS）と呼ばれている．図2は，I/Oなどの部分を除いたCeNSSの構成イメージを示したものである．ノードは，32個のCPUから成るSMP（Symmetric Multi Processors）を構成し，1ノードは64GBの共有メモリ空間を有する．ここで，DTUとは，Data Transfer Unitの略で，結合ネットワークにデータを送り出す/受け取る論理上の装置を表す．DTUあたり，16プロセスを同時に処理することができる．表1に，CeNSSの構成諸元を示した．4ノードで1筐体を構成しており，計算筐体は全部で14筐体，ノード数では56ノードある．筐体は，富士通製PRIMEPOWER HPC2500である．CPUは，SPARC64 Vを採用し，ピークで5.2GFLOPSの性能と2MBのオンチップL2キャッシュを有する．こうした構成のシステムはSMPクラスタと呼ばれることがあり，どのCPUからみても厳密に互いに対称な配置を有している．図3は，CeNSSにおけるFortranの並列プログラミング体系を示したものである．ノード内では，コンパイラによる自動並列またはOpenMPあるいはそれらの混在によるスレッド並列を用い，ノード間では，MPIまたはデータ並列言語の一種であるXPFortran(XPF)によるプロセス並列を組み合わせることにより，いわゆるハイブリッド並列のスタイルを標準として採用している．ここで，ノード内のスレッド並列は必須ではなく，MPIやXPFortranによるプロセス並列だけの並列プログラミングも可能であることに注意する．詳細は，文献2を参照されたい．

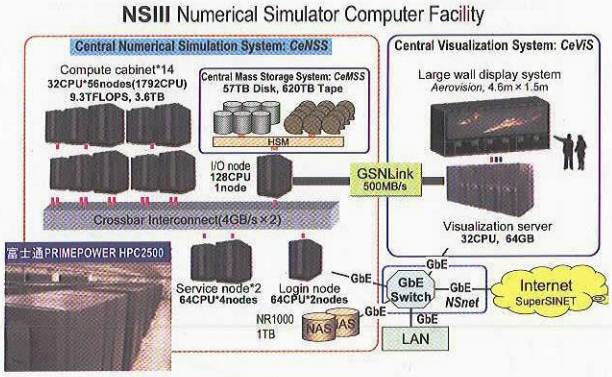


図1 JAXA 数値シミュレータ III（NS-III）の構成概要

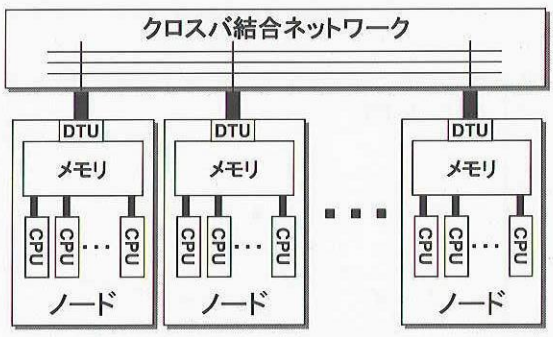


図2 CeNSS の構成イメージ

表1 CeNSS の諸元

| | |
|---|---|
| 総合演算処理性能 | 9.3TFLOPS |
| 総メモリ量 | 3.6TB |
| 計算用ノード数 | 56 |
| ノード内 CPU 数 | 32 |
| ノード内メモリ量 | 64GB |
| ノード構成 | SMP |
| 計算用 CPU 総数 | 1,792 |
| CPU タイプ | SPARC64 V |
| CPU ピーク性能 | 5.2GFLOPS |
| L2 キャッシュ | 2MB オンチップ |
| 結合ﾈｯﾄﾜｰｸﾄﾎﾟﾛｼﾞ | 単段クロスバ |
| 結合ﾈｯﾄﾜｰｸ性能 | 4GB/s×2 |

| ノード内 | ノード間 |
|---|---|
| XPFortran | |
| 自動並列 | XPFortran |
| OpenMP | |
| 自動並列+OpenMP | |
| 自動並列 | MPI |
| OpenMP | |
| 自動並列+OpenMP | |
| MPI | |

図3 CeNSS における Fortran プログラミング体系

表2 性能評価した JAXA 並列 CFD コード

| Code (Name) | Application | Simulation Mmdel | Numerical method | Parallel strategy | Lang. |
|---|---|---|---|---|---|
| P1 (LES) | Aircraft | LES | FDM | OpenMP + MPI | F77 |
| P2 (HJET) | Combustion | DNS | FDM w. Chemistry | OpenMP + MPI | F77 |
| P3 (CHANL) | Turbulence | DNS | FDM with FFT | OpenMP + XPF | F77 |
| P4 (HELI) | Helicopter | URANS | FDM w.Overlapped | AutoPara + XPF | F77 |
| P5 (UPACS) | Aeronautics | RANS | FVM w.Multiblock | MPI | F90 |
| P6 (JTAS) | Aeronautics | RANS | FVM w.Unstructured | MPI | F77 |

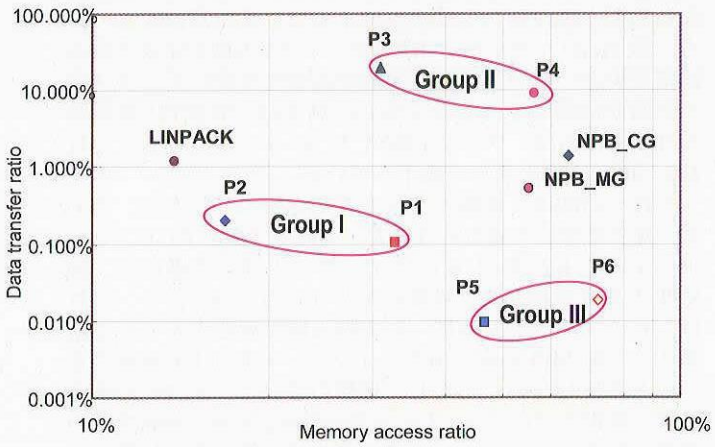


図4 JAXA 並列 CFD コードの特性

## 3．JAXA 並列 CFD コードの性能評価

表 2 に，本報告で性能推定用に評価した 6 本の JAXA の主に航空関係で実際に使われている並列 CFD コード[3)~8)]の概要を示す．このうち，コード P2，P3 が学術系の課題（P2:燃焼流，P3:平行平板間乱流）を解析するためのものであり，あとの 4 本は工学系のコードである．基礎式は，いずれのコードも Navier-Stokes 方程式であり，格子形状によってメモリアクセスパターン（ローカル or グローバル or 連続 or リスト）や転送パターンが，流体以外の部分を解く解かない等で演算密度が異なる．図 4 は，横軸にメモリコスト，縦軸に通信コストを取って，各コードの特性をプロットしたものである．ここで，メモリコスト=メモリアクセス時間/CPU 時間，通信コスト=通信時間/経過時間として，プロファイラ等で採取したデータから持って来たものである．無論，同じ CPU 数でも使用したスレッド数，プロセス数の組み合わせによって，あるいは問題サイズなどによってプロットされる位置は多少ずれるが配置は大きくは変わらない．プロットした値は，プロセス数はすべて 4 で統一して測定したものである．これより，6 本の CFD コードは，計算処理中心のグループ I（コード P1, P2），通信コストが多い（10%以上）グループ II（コード P3, P4），メモリアクセスが多いグループ III（コード P5, P6）の 3 グループにほぼ分類できることがわかった．ちなみに，ベンチマークとして有名な LINPACK 及び NAS Parallel Benchmarks の MG と CG を，参考までにプロットした．

まず，各コードに対する CeNSS 上での並列性能の実測結果を示す．問題サイズ（空間の格子分割数）を一定にして，スレッド数を固定し，プロセス数を変化させることにより，

経過時間を測定した．格子分割数は，できるだけ実問題に近いサイズとした．また，他のジョブからの影響を避けるために，他のジョブが一切ない状態で測定した．

図 5(a)は，グループ I のコード P1 の測定結果である．横軸にプロセス数，縦軸には，1 プロセス×1 スレッドのときの性能を基準(=1)としたときの性能比を取ってあり，何本かの線は，スレッドを 1,2,4,8,16 と変化させたものに相当する．図からわかるように，グループ I の場合，メモリアクセス，通信ともに少ないので，プロセスに対してもスレッドに対してもスケーラビリティは良好である．コード P2 も同様の特性を示す．図 5(b)に，グループ II のコード P3 の測定結果を示す．縦軸は，28 プロセス×1 スレッドの値を基準としたときの性能比である．通信が多いために，プロセス性能の直線性は良くなく，プロセス数が多くなると性能曲線は傾きは寝てくる．一方で，スレッド並列については，プロセス数一定のラインでみると，スレッドが多くなっても一定の割合で性能が向上しているのがわかる．一方，図 5(c)は，グループ III のコード P5 の測定結果である．通信は少ないのでプロセス性能の直線性は良いが，多スレッド時の性能向上は大きくない．例えば，512CPU 使用時の性能向上比は，スレッド数が 2，4，8 と多くなるにつれ，370，300，220 と大きく低下している．

これらの図は，横軸にプロセス数を取っているが，CPU 数一定で整理した場合の性能を，コード P1 と P3 でプロットしてみたものを図 6 に示す．コード P1 の場合は，純 MPI の場合が最も良い性能を示しているのがわかる．これは，ハイブリッド並列についての一般に報告されている結果 [1]，「純 MPI 並列の方がハイブリッド並列より性能が良い」と矛盾しない．しかし，通信の多いコード P3 の場合には，特定のプロセス×スレッドの組み合わせ（448CPU の場合 56 プロセス×8 スレッド）のときに最も良い性能を示し，純 MPI の場合に比べ，2 割ほど高い性能を示している．これは，P3 のようなコードの場合，プロセス×スレッドの組み合わせを適切に選ぶことにより，プログラムに手を加えることなく性能向上を図ることができることを意味している．

以上から，CeNSS においては，図 5 に示したようにコードによってその並列性能は大きな差異を示すが，そのパターンは図 4 のようなコードの特性（メモリコスト，通信コストなど）によってうまく整理できるといえる．

## 4．拡張アムダールの法則による性能推定

### 4.1 アムダールの法則

並列システムの性能を表す法則の一つにアムダールの法則がある．いま，あるプログラムにおいて，並列処理できる部分の割合（並列化率）を $a$ とし，$n$ 台の CPU を用いて得られる性能向上率を $S(n)$ とすると，

$$S(n) = T_{serial}/T_{parallel} \qquad (1)$$

ただし，$T_{serial}$，$T_{parallel}$ は，それぞれ逐次実行，並列実行時の CPU 時間をあらわし，

$$T_{parallel} = T_{serial} \times \{(1 - a) + a/n\} \qquad (2)$$

の関係がある．もし，$a = 1$ なら $T_{parallel} = T_{serial}/n$，ゆえに $S(n) = n$ となり，並列性能は CPU 数の増加とともに完全にリニアに上昇する．また，$n \to \infty$ とすると $S(\infty) \to 1/(1-a)$ であり，これは並列化率に応じて性能向上に上限があることを意味している．たとえば，$a = 0.95$ の場合 $S(\infty) = 20$，すなわち性能向上率はたかだか 20，よって，20CPU 以上使うのは無駄ということになる．ただし，一般的には，$n$ が大きくなると $a$ も上昇するので，状況はそれほど悲観的ではないことに注意する．

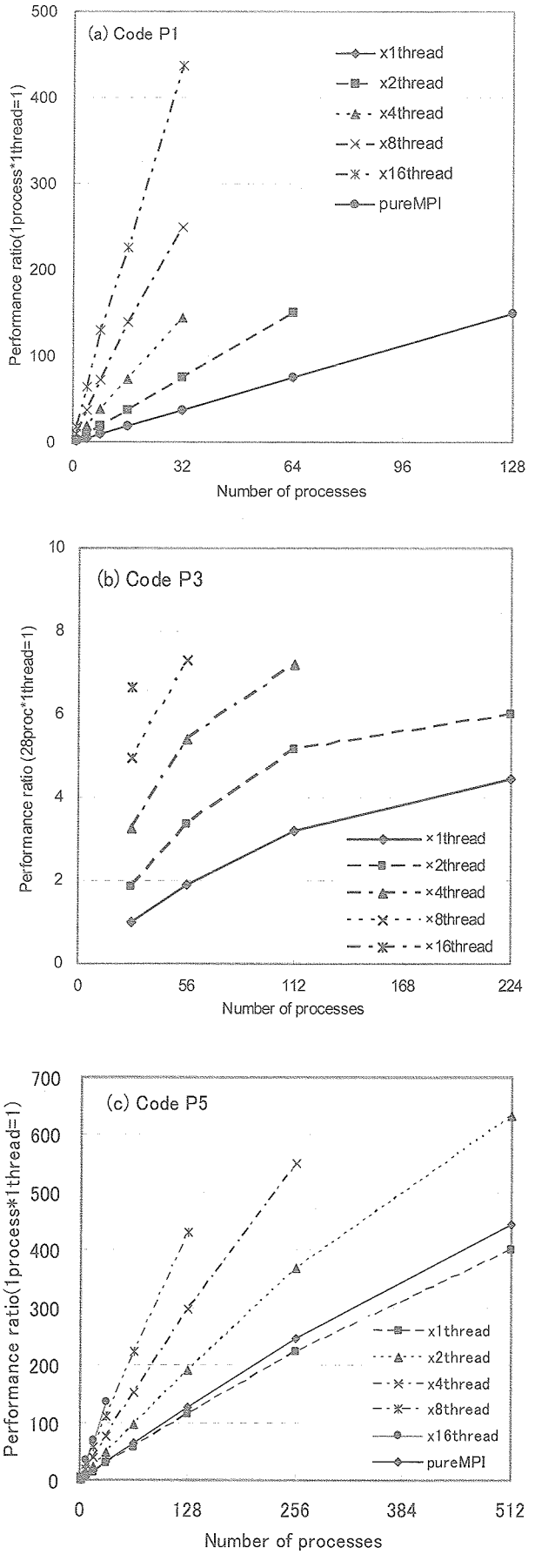


図 5　JAXA 並列 CFD コードの性能

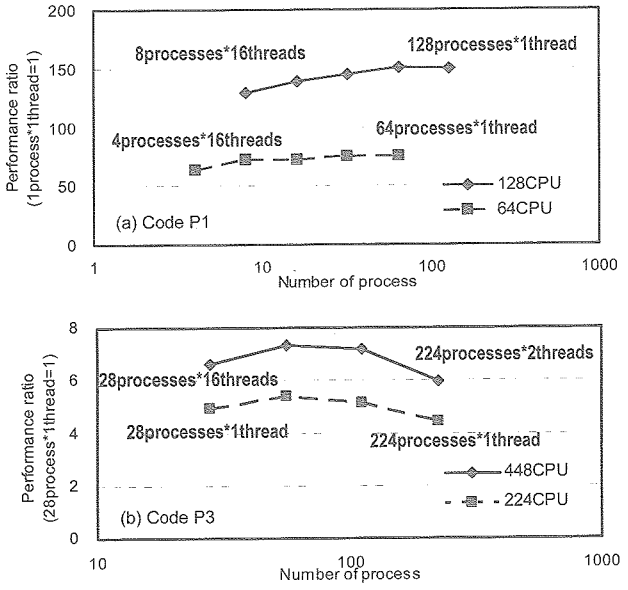


図 6　CPU 数一定時の並列性能の比較

### 4.2 ハイブリッド並列におけるアムダールの法則の拡張とその検証（その 1）

ハイブリッド並列における並列形態には，プロセス並列とスレッド並列が存在するので，上記の一般的なアムダールの法則は直接的には適用できない．いま，プロセス並列の並列数を $n_p$，並列化率を $a_p$，スレッド並列の並列数を $n_t$，並列化率を $a_t$ とする．表 3(a)は，コード P1 の並列性能の実測値の一部を表にしたものである．4 プロセス×1 スレッドの場合の性能を基準(=1)としたときの性能比を示してある．プロセス並列化率 $a_p$ は，スレッド 1 の場合の性能向上比（第 1 行）から，スレッド並列化率 $a_t$ は，プロセス 1 の場合の性能向上比（第 1 列）から，式(1)(2)により導かれる関係 $a = (1\text{-}1/S)(1\text{-}1/n)$ を用いて求めることができ，これより平均の $a_p = 1.018$，$a_t = 0.994$ と求まる．いま，ハイブリッド並列におけるアムダールの法則として，通常のアムダールの法則(1)(2)の自然な拡張として，

$$S(n) = T_{serial}/T_{hybrid} \qquad (3)$$

ただし，

$$T_{hybrid} = T_{serial} \times \{(1 - a_p) + a_p/n_p\} \times \{(1 - a_t) + a_t/n_t\} \qquad (4)$$

を考える．ここに，$(1 - a_p) + a_p/n_p$ はプロセス並列による加速分，$(1 - a_t) + a_t/n_t$ はスレッド並列による加速分をあらわす．上記の $a_p$ 及び $a_t$ を式(3)(4)に代入して，性能向上比を推定したのが表 3(b)である．表 3(c)は，表 3(a)の実測値と表 3(b)の推定値をもとに，推定値/実測値を示したものである．塗りつぶした欄の値を除きほとんどの値が 1 に近いことから，このコード P1 の場合は，拡張アムダールの法則(3)(4)により，性能推定が可能であることがわかる．なお，高並列で誤差が大きくなるのは，このコード P1 の場合，データアクセスがオンキャッシュになり，並列数以上に性能が出てしまっているからである．

表 4 は，コード P5 の場合の実測値と拡張アムダールの法則による推定値を比較したものである．表 4(a)は，実測値であるが，これより平均の $a_p = 0.995$，$a_t = 0.826$ と求まる．この値から式(3)(4)を用いて性能向上比を推定したのが表 4(b)，実測値と推定値を比較したものが表 4(c)である．このコード P5 の場合も，推定値と実測値はすべてのレンジで比較的良く一致している．

表 3　拡張アムダールの法則の検証（コード P3）

(a)　性能向上比の実測値

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 2.11 | 4.23 | 8.83 | 17.40 |
| 2 | 2.00 | 4.21 | 8.14 | 17.42 | 34.06 |
| 4 | 3.93 | 8.30 | 15.97 | 34.37 | - |
| 8 | 7.60 | 16.06 | 28.61 | 67.15 | - |
| 16 | 14.04 | 29.14 | 54.75 | 104.56 | - |

（数字は，4 プロセス×1 スレッドの値を基準）

(b)　性能向上比の推定値

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 2.04 | 4.28 | 9.44 | 23.75 |
| 2 | 1.99 | 4.06 | 8.50 | 18.75 | 47.20 |
| 4 | 3.93 | 8.03 | 16.80 | 37.04 | 93.22 |
| 8 | 7.66 | 15.66 | 32.78 | 72.29 | 181.92 |
| 16 | 14.61 | 29.83 | 62.53 | 137.90 | 347.01 |

(c)　推定値と実測値の比較（推定値/実測値）

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 0.97 | 1.01 | 1.08 | 1.36 |
| 2 | 1.00 | 0.97 | 1.04 | 1.08 | 1.36 |
| 4 | 1.00 | 0.97 | 1.05 | 1.08 | - |
| 8 | 1.01 | 0.98 | 1.15 | 1.08 | - |
| 16 | 1.04 | 1.03 | 1.14 | 1.32 | - |

表 4　拡張アムダールの法則の検証（コード P5）

(a)　性能向上比の実測値

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 2.00 | 3.96 | 7.70 | 13.81 |
| 2 | 1.66 | 3.29 | 6.55 | 12.66 | 21.78 |
| 4 | 2.62 | 5.21 | 10.25 | 18.92 | - |
| 8 | 3.83 | 7.65 | 14.83 | - | - |
| 16 | 4.69 | - | - | - | - |

（数字は，32 プロセス×1 スレッドの値を基準）

(b)　性能向上比の推定値

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 1.99 | 3.94 | 7.72 | 14.83 |
| 2 | 1.70 | 3.39 | 6.71 | 13.14 | 25.27 |
| 4 | 2.63 | 5.23 | 10.35 | 20.27 | 38.97 |
| 8 | 3.60 | 7.17 | 14.20 | 27.82 | 53.47 |
| 16 | 4.43 | 8.81 | 17.44 | 34.18 | 65.70 |

(c)　推定値と実測値の比較（推定値/実測値）

| スレッド比＼プロセス比 | 1 | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| 1 | 1.00 | 1.00 | 1.00 | 1.00 | 1.07 |
| 2 | 1.03 | 1.03 | 1.02 | 1.01 | 1.16 |
| 4 | 1.00 | 1.00 | 1.01 | 1.07 | - |
| 8 | 0.94 | 0.94 | 0.96 | - | - |
| 16 | 0.94 | - | - | - | - |

表 5 拡張アムダールの法則の検証（コード P3）

(a) 性能向上比の実測値

| | | プロセス比 | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 4 | 8 | 16 |
| スレッド比 | 1 | 1.00 | 1.90 | 3.21 | 4.43 | - |
| | 2 | 1.85 | 3.35 | 5.14 | 5.98 | - |
| | 4 | 3.25 | 5.41 | 7.19 | - | - |
| | 8 | 4.94 | 7.32 | - | - | - |
| | 16 | 6.63 | - | - | - | - |

（数字は，28 プロセス×1 スレッドの値を基準）

(b) 性能向上比の推定値

| | | プロセス比 | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 4 | 8 | 16 |
| スレッド比 | 1 | 1.00 | 1.85 | 3.20 | 5.04 | 7.09 |
| | 2 | 1.84 | 3.40 | 5.90 | 9.30 | 13.08 |
| | 4 | 3.19 | 5.89 | 10.20 | 16.09 | 22.63 |
| | 8 | 5.02 | 9.27 | 16.06 | 25.34 | 35.63 |
| | 16 | 7.05 | 13.01 | 22.53 | 45.55 | 50.00 |

(c) 推定値と実測値の比較（推定値/実測値）

| | | プロセス比 | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 4 | 8 | 16 |
| スレッド比 | 1 | 1.00 | 0.97 | 1.00 | 1.14 | - |
| | 2 | 1.00 | 1.02 | 1.15 | 1.55 | - |
| | 4 | 0.98 | 1.09 | 1.42 | - | - |
| | 8 | 1.02 | 1.27 | - | - | - |
| | 16 | 1.06 | - | - | - | - |

表 6 コード P3 における通信コストの影響

| | プロセス数 | | | |
|---|---|---|---|---|
| | 28 | 56 | 112 | 224 |
| 実測性能向上比 | 1.00 | 1.90 | 3.21 | 4.43 |
| 並列化率 | - | 0.946 | 0.916 | 0.885 |
| 経過時間（秒） | 464.87 | 247.66 | 149.69 | 110.29 |
| 通信時間（秒） | 31.65 | 28.78 | 34.76 | 46.14 |

（並列化率は 28 プロセスを基準に算出）

このことから，通信コストの少ないグループ I やグループ III のコードについては，拡張アムダールの法則(3)(4)により性能推定が可能であることがわかる．その理由としては，基本的に通信コストが少ないことの他に，式(4)が示しているように，プロセスとスレッドの並列効果に依存性がないことが考えられる．実際，多くの CFD コードでは，多数の多重 Do ループ構造を持っており，外側の Do ループに対してプロセス並列を，内側のループに対してはスレッド並列を適用するハイブリッド戦略を採っていることが強く関係していると考えられる．データは省略するが，コード P4 の場合，多重 Do ループに依存性があるためにこの拡張アムダールの法則による性能推定はうまく行かない．

一方，コード P3 の場合の結果を表 5 に示す．この場合，平均の $a_p = 0.916$，平均の $a_t = 0.915$ である．表 5(c)によれば，塗りつぶした欄の値のように，プロセス数×スレッド数の大きいところで推定の誤差が大きくなっている．

### 4.3 ハイブリッド並列におけるアムダールの法則の拡張とその検証（その 2）

そこで次に，コード P3 における性能推定誤差の原因とアムダールの法則の改良（高精度化）を考える．いま，プロセス数が変化したときの並列化率を調べてみると，表 6 上段のようになり，並列化率は一定ではなく，プロセス数の増加とともにかなり低下しているのがわかる．これは，プロセス数とともに増加する通信コスト等が存在するためである．プロファイラを使いコストの内訳を測定した結果，表 6 下段に示すようにプロセス数の増加に伴う通信コストの増大を実際に確認した．

そこで，アムダールの法則(1)(2)に対して，プロセス並列における通信コストの影響を考慮する高精度化を考える．いま，プロセスの並列化率 $a_p$，プロセス数 $n_p$ に加えて，通信の影響として，プロセス数によらずコスト一定の通信の割合を $c_t$，袖転送のようなプロセス数にコストが比例する通信の割合を $c_n$ とすると，アムダールの法則(1)(2)の自然な拡張として，

$$S(n) = T_{serial}/T_{parallel} \tag{5}$$

ただし，

$$T_{parallel} = T_{serial} \times \{(1 - a_p - c_t - c_n) + a_p/n_p + (c_t + c_n \times n_p)\} \tag{6}$$

という関係を考えることができる．コード P3 におけるそれぞれのコストをプロファイラによって調べてみると，28 プロセスの場合，$a_p = 0.925$，$c_t = 0.057$，$c_n = 0.005$，$1\text{-}a_p\text{-}c_t\text{-}c_n = 0.013$ となり，6%強の実際に無視できない通信コストが存在する*．式(5)(6)により求めた推定性能向上比と実測値とを比較したのが表 7 であり，(1)(2)によるものと比べると，多プロセスにおける性能推定の精度は向上している．さらに，実測範囲以上の多プロセスの場合の性能向上比を推定してみると，表 7 にあるように，896 プロセス以上では，通信コストの影響で性能向上比が低下する傾向が現れている．

この結果を利用して，ハイブリッド並列における拡張アムダールの法則を高精度化してみると，通信の部分はスレッド並列による加速の影響は受けないことから，

$$S(n) = T_{serial}/T_{hybrid} \tag{7}$$

ただし，

$$T_{hybrid} = T_{serial} \times [\{(1 - a_p - c_t - c_n) + a_p/n_p\} \times \{(1 - a_t) + a_t/n_t\} + (c_t + c_n \times n_p)] \tag{8}$$

とすることができる．表 8 は，コード P3 に対して式(7)(8)による推定値と実測値を比較したものであるが，両者はすべてのレンジでよく合致しており，表 5(c)と比べても高プロセス×高スレッドの場合の推定精度は改善されているのがわかる．表 9 は，表 8(a)を CPU 数一定で整理したものである．図 6(b)で示した特定のプロセス×スレッドで性能が高くなる挙動がよく再現されており，高精度拡張アムダールの法則(7)(8)が通信の多いコード P3 の場合の性能推定に有効であることを示している．

*ちなみにこれはプロセス別の演算コストを合計した，いわば非並列状態での比率であり，プロセス数に応じた通信コストの比率は式(6)をもとに算出することができる．例えば,4 プロセス並列時の実行時間比は，$T_{parallel}/T_{serial} = 0.013 + 0.925/4 + 0.057 + 0.005 \times 4 = 0.321$ であり，それに含まれる通信の影響は，$c_t + c_n \times n_p = 0.057 + 0.005 \times 4 = 0.077$ であるから，通信コストの比率は，$0.077/0.321 \approx 0.24$ となり，これは図 4 で示した特性図の実測値とも概ね一致する．

表7 高精度アムダールの法則によるプロセス性能推定（コードP3）

| | プロセス数 | | | | | | |
|---|---|---|---|---|---|---|---|
| | 28 | 56 | 112 | 224 | 448 | 896 | 1,792 |
| 実測性能向上比 | 1.00 | 1.90 | 3.21 | 4.43 | - | - | - |
| (1)(2)による推定性能向上比 | 1.00 | 1.85 | 3.20 | 5.04 | 7.08 | 8.88 | 10.17 |
| (5)(6)による推定性能向上比 | 1.00 | 1.84 | 3.11 | 4.43 | 4.81 | 3.86 | 2.47 |

表8 高精度拡張アムダールの法則の検証（コードP3）

(a) 性能向上比の推定値

| | | プロセス比 | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 4 | 8 | 16 |
| スレッド比 | 1 | 1.00 | 1.84 | 3.11 | 4.43 | 4.81 |
| | 2 | 1.84 | 3.22 | 4.94 | 6.14 | 5.77 |
| | 4 | 3.18 | 5.12 | 7.00 | 7.60 | 6.41 |
| | 8 | 4.99 | 7.29 | 8.84 | 8.62 | 6.78 |
| | 16 | 6.97 | 9.23 | 10.18 | 9.24 | 6.99 |

(b) 推定値と実測値の比較（推定値/実測値）

| | | プロセス比 | | | | |
|---|---|---|---|---|---|---|
| | | 1 | 2 | 4 | 8 | 16 |
| スレッド比 | 1 | 1.00 | 0.97 | 0.97 | 1.00 | - |
| | 2 | 1.00 | 0.96 | 0.96 | 1.03 | - |
| | 4 | 0.98 | 0.95 | 0.97 | - | - |
| | 8 | 1.01 | 1.00 | - | - | - |
| | 16 | 1.05 | - | - | - | - |

表9 高精度拡張アムダールの法則による性能推定（コードP3）

| | | CPU数（プロセス数×スレッド数） | | | |
|---|---|---|---|---|---|
| | | 112 | 224 | 448 | 896 |
| スレッド数 | 1 | 3.11 | 4.43 | 4.81 | 3.86 |
| | 2 | 3.22 | 4.94 | 6.14 | 5.77 |
| | 4 | 3.18 | 5.12 | 7.00 | 7.60 |
| | 8 | 2.99 | 4.99 | 7.29 | 8.84 |
| | 16 | 2.63 | 4.52 | 6.97 | 9.23 |

## 5. まとめ

本報告では，JAXAにおける並列CFDコードの性能測定結果を示すとともに，コードの特性と並列性能の関係について論じた．また，アムダールの法則を拡張したハイブリッド並列における簡易な性能推定法を提示し，その推定精度を検証した．JAXAのハイブリッド並列CFDコードの特性を分析した結果，通信が少ない場合は拡張アムダールの法則，通信が多い場合は高精度拡張アムダールの法則で性能向上比の推定が可能であることがわかった．ここで示した性能推定法は，特殊なパラメータを引用しているわけではないので，JAXAの並列システムに限らず，一般の並列システムに適用できるものである．

しかし，今回実施したような系統的性能評価は一般には困難と考えられるから，拡張アムダールの法則(3)(4)や(7)(8)の基本になっている並列化率や通信コストを如何に簡便に見積もるかがこの推定法の鍵でありそれがまた今後の課題でもある．例えば，コードP1のような通信量が少なく線形の性能の場合には，プロセス×スレッドの組み合わせとして，16×1, 1×16のように，2ケースでプロセス並列化率とスレッド並列化率を採取すれば，(3)(4)により精度良い性能推定が可能である（確認済み）．しかし，コードP3, P4のように通信量が多い場合には，(7)(8)を使う必要があり，しかも，その場合には，プロセス数に比例するかどうか等の通信の中身まで把握する必要がある．通信量の全体はプロファイラで知ることができるが，通信の中身を簡単に測る方法については今後の検討と考えている．

本研究を進めるにあたり，保守時間等の合間をぬってCeNSSを使わせていただいた．ここに記して謝意を表する．

## 参考文献


1) Cappelo, F. and Etiemble, D.: MPI versus MPI+OpenMP on IBM SP for the NAS Benchmarks, *Proc. SC'00*, Dallas, USA (Nov. 2000).
2) Matsuo, Y., Tsuchiya, M., Aoki, M., Sueyasu, N., Inari, T. and Yazawa, K.: Early Experience with Aerospace CFD at JAXA on the Fujitsu PRIMEPOWER HPC2500, *Proc. SC'04*, Pittsburgh, USA (Nov. 2004).
3) 松尾裕一：差分法による翼まわり流れのLES，第12回数値流体力学シンポジウム講演論文集，pp.153-154 (Dec. 1998).
4) 溝渕泰寛，新城淳史，小川哲：CeNSSを用いた水素噴流浮き上がり火炎詳細シミュレーション，航空宇宙数値シミュレーション技術シンポジウム2004論文集，JAXA特別資料SP-04-012, pp.202-207 (Mar. 2005).
5) 阿部浩幸，松尾裕一：平行平板間乱流の大規模直接数値シミュレーション，航空宇宙数値シミュレーション技術シンポジウム2004論文集，JAXA特別資料SP-04-012, pp.21-26 (Mar. 2005).
6) 近藤夏樹，青山剛史，齊藤茂：重合格子法を用いたロータ/胴体干渉の計算，航空宇宙数値シミュレーション技術シンポジウム2003論文集，JAXA特別資料SP-03-002, pp.232-237 (Mar. 2004).
7) Takaki, R., Yamamoto, K., Yamane, T., Enomoto, S. and Mukai, J.: The Development of the UPACS CFD Environment, *High Performance Computing, Proc. ISHPC2003*, LNCS 2858, pp.307-319 (Oct. 2003).
8) 村山光宏，山本一臣：非構造格子法を用いた航空機高揚力装置周りの流れ場解析の精度検証，航空宇宙数値シミュレーション技術シンポジウム2004論文集，JAXA特別資料SP-04-012, pp.82-86 (Mar. 2005).
9) Dongarra, J., Bunch, J., Moler, C. and Stewart, G.W.: LINPACK User's Guide, SIAM Philadelphia, PA (1979).
10) Baily, D., Harris, T., Saphir, W., van der Wijngaart, R., Woo, A. and Yarrow, M.: The NAS Parallel Benchmarks 2.0,NAS Technical Report NAS-95-020, NASA Ames Research Center (1995).