2.3 非構造格子 Euler/Navier-Stokes ソルバ JTAS のスレッド並列最適化

坂下雅秀(宇宙航空研究開発機構)

概　　要

3 次元ハイブリッド非構造格子有限体積法 Euler／Navier-Stokes ソルバ JTAS（JAXA Tohoku university Aerodynamic Simulation code）は、元来、ベクトル計算機用に開発されたものであり、スレッド並列による実行が可能である。しかし、テストデータによる性能測定において、時間積分計算の部分で、8 スレッド実行によるスレッド並列加速率が約 5 倍と、理論値（8 倍）の 7 割を下回る性能しか得られていなかった。そこで、全体性能の向上とスレッド並列化の最適化を目的とした変更を加え、JTAS スレッド並列版を開発した。このスレッド版について、同様にテストデータによる性能測定を行った結果、全体の性能が約 1.5 倍向上し、時間積分計算部分で、8 スレッド実行によるスレッド並列化加速率が約 6.2 倍と、理論値の 7 割を越える性能が得られることが確認された。

1.　はじめに

現在、宇宙航空研究開発機構(JAXA)では、次世代超音速機技術の基礎研究として小型超音速実験機（NEXST－1）に関するプロジェクトが進められている[1][2]。このプロジェクトにおいては、複雑な形状の回りにおける剥離や再付着を伴う複雑な流れ場に対するCFD（Computational Fluid Dynamics）解析技術が求められている。このような解析には非構造格子法（Unstructured Grid Method）が良く用いられる。JAXAにおいては、非構造格子ソルバとして、主にJTASが用いられている[3][4][5]。JTASは、東北大学で開発されたTAS(Tohoku university Aerodynamic Simulation code)[6]をもとにJAXAに導入されているCeNSS(Central Numerical Simulation System)と呼ばれる大規模SMP（Symmetric Multiple Processor）クラスタシステム（富士通製 PRIMEPOWER HPC2500）に適合するように若干の変更が加えられたコードであり、オリジナルのTASと区別する意味でJTASと呼ばれている。

JTAS は、CeNSS 向けに変更が加えられているものの、その変更は配列の次元入れ替え等限定的なものであり、CeNSS の性能を十分有効に活用出来ていないという問題があった。そこで、FORTRAN コンパイラによる自動並列化のより効率的な活用を促進することで CeNSS に対する適合性を向上させることを目的として、より内容に踏み込んだ変更を行った。また、テストデータを用いた性能測定を実施し、変更による性能向上を確認した。

2.　JTAS 概要

性能評価には、支配方程式として３次元Euler方程式を用いた。JTASではこれを、空間方向にはセル節点有限体積法により、時間方向にはEuler陰解法により離散化し、時間積分にはLU-SGS陰解法[10][11]を用いて計算する。流束の評価は、HLLEW法[8]により行われる。高次精度差分における制限関数としては、Venkatakrishnanの制限関数[9]が用いられている。

JTASは、元来ベクトル計算機用に開発されているため、LU－SGS法の適用にあたっては非構造格子に適用可能な超平面（Hyper Plane）を構成し再帰参照を回避している[10][11]。ところで、有限体積法において、流束ベクトルは、各検査体積を構成する面ごとに求める必要がある。一方で、ある面は異なる二つの検査体積の境界を構成するのであるから、その面を通る流束はその二つの検査体積に対して同じ量でかつ符号が反対であるように寄与することになる。従って、流束ベクトルを検査体積ごとにそれを構成する全ての面に対して計算することは、流束ベクトルを二重に計算することとなり効率的ではない。面ごとに計算すれば流束ベクトルの計算を最小限に押さえることが出来る。ここで、セル節点体積法の場合、検査体積がその唯一含む節点の番号で指定されるのと同様に、面はそれが唯一含む辺の番号で指定されるので、実際のプログラムにおける流束の計算は、辺IEが含む両端の節点の番号N1 及びN2 を保持する配列の名前を"NEDG2ND"とした場合、例えばリスト 2.1 のようになる。

リスト 2.1　流束計算の例

```
    DO 100 IE = 1, Nedges
       N1=NEDG2ND(1,IE)
       N2=NEDG2ND(2,IE)
       HLLEW=FLUX_FUNC(Same arguments required)
       FLUX(N1)=FLUX(N1)+HLLEW
       FLUX(N2)=FLUX(N2)-HELLW
100 CONTINUE
```

ここで、このDOループは配列"FLUX"に対して再帰参照になっていることに注意されたい。なぜならば、検査体積は複数の面から構成されているため、異なる面（辺）IEにおいて同じ検査体積番号（節点番号）がN1 又はN2 に表れるからである。

JTASでは、この再帰参照を回避しベクトル計算を行うため、色分け（Coloring）によるグループ化の手法が用いられている。これは、ある検査体積（節点）

に含まれる全ての面（辺）は必ず別の色を持つように前もって色分けしておき、DOループ内では同じ色を持った面（辺）のみを計算することにより、再帰参照を避ける方法である。この場合、実際のプログラムは、例えばリスト 2.2 に示すような二重ループになる。外側のDOループが全ての色に対する処理のループであり、内側のDOループがその内のある一つの色に対する処理を行うDOループである。色分けを適切に行うことにより、内側のループで一度に処理される面（辺）は必ず異なる検査体積（節点）を構成するものとなり、再帰参照が回避されベクトル化される[12]。

リスト 2.2　流束計算のベクトル化例

```
      DO 100 IC = 1, MAX_Colors
*VOCL LOOP,NOVREC
      DO 110 IE = 1, Num_edges(IC)
         N1=NEDG2ND(1,IE,IC)
         N2=NEDG2ND(2,IE,IC)
         HLLEW=FLUX_FUNC(Some arguments required)
         FLUX(N1)=FLUX(N1)+HLLEW
         FLUX(N2)=FLUX(N2)-HELLW
  110 CONTINUE
  100 CONTINUE
```

同様のベクトル化手法は、速度、圧力、密度及び温度の勾配（Gradient）の計算（以下、単に勾配の計算という）、制限関数の計算、LU－SGS法における計算の一部においても使用されている。但し、勾配の計算においては、計算が面（辺）ごとではなく要素毎に行われることが異なる。

JTASでは、MPIを利用したプロセス並列化もなされている。プロセス並列化を行うにあたっては、まず計算に先立って各プロセスに割り当てるために格子空間を領域分割し、この分割された領域をそれぞれのプロセスが分担して計算する[13]。

3.　計算性能最適化のための変更

全体の性能及びスレッド並列における性能向上を目的に JTAS に加えた変更内容は、以下の二点である。

(1) LU－SGS 法における節点番号の付け替え

(2) 色分けの削除及び DO ループの分割

以下、この変更を加えた JTAS をスレッド版 JTAS、または単にスレッド版という。より具体的な変更内容を以下に示す。

3.1.　LU-SGS 法における節点番号の付け替え

JTAS では非構造格子に LU－SGS 法を適用するために超平面が構成されている。ところが、節点における各物理量等を配列に保存する際には、格子生成時に付されたオリジナルの節点の番号でインデックスされる場所に保存されている。このような方法は、或る一つの超平面内に存在する節点の番号が不連続であるため、効率的なメモリアクセスを疎外する要因となることが予想された。そこで、LU－SGS 法の計算を行う部分では、ハイパー面を考慮した節点番号の付け替えを行うこととし、新たな節点番号として超平面内でオリジナルの節点番号の昇順に連続な番号を与えた。この際、LU－SGS 法に関係する部分以外では従来通りの番号付けとし、LU－SGS 法で必要となる保存量ベクトル等のデータは、従来の番号付けで保存されている配列から新たな番号付けで保存される配列に一旦コピーした後 LU－SGS 法の計算を行い、新しい時間ステップでの値を計算する際に従来の番号付けの配列に戻す方法をとった。

3.2.　色分けの削除及び DO ループの分割

オリジナルの JTAS は、既にベクトル化されているため一切の変更を加えることなしに、FORTRAN の持つ自動並列化機能によりスレッド並列化することが可能である。ところで、ベクトル化は基本的に最内側 DO ループに対してなされる。一方で、スレッド並列化では、スレッド生成のオーバーヘッドをなるべく小さくするために、スレッド生成回数の少ない、より外側の DO ループで並列化することが望ましい。色分けによるベクトル化では、例えばリスト 2.2 において内側の DO ループである DO 110 がベクトル化、即ちスレッド並列化されることとなり、スレッド生成のオーバーヘッドによる性能低下が予想された。加えて、スレッド並列化される DO ループの回転数は、生成されたスレッドの中でなるべく多くの演算が行われるように、ベクトル化における場合同様なるべく多い方が望ましいが、色分けによるベクトル化では、一度に計算されるのは一つの色に属する辺（勾配の計算の場合要素）のみであり、全ての辺を一度に処理するのに比べて性能が低下することが予想される。一方で、全ての辺について同時に計算することにすれば、リスト 2.1 に示すように DO ループは一重ループとなり、外側かつ回転数の多い理想的なループの構成となるが、再帰参照を含むため、ベクトル化もスレッド並列化も行うことが出来ない。

そこで、リスト 3.1 に示すように色分けの削除をすると共にDOループの分割を行うことで、色分けによる方法で問題になると予想される点の改善を図った。リスト 3.1 は、流束ベクトルの計算を簡略化した例であり、DO 100 において辺ごとに計算された流束ベクトルは、一旦、作業用配列"EDG_WK"に辺ごとの値として保存された後、DO 110 において、節点（検査体積）ごとの配列"FLUX"に足し込まれている。ここで、DO 100 における配列"EDG_WK"及びDO 110 における配列"FLUX"は、インデックス参照ではなく直接参照となっていることに注意されたい。これにより、メモリアクセスの効率化も期待される。

尚、この変更は、LU－SGS法の計算に対しては適用しなかった。これは、色分けを行った方が良い性能が得られたためである（「5.5 LU－SGS法の計算における測定結果及び評価」参照）。

リスト 3.1　流束計算の変更例

```
    DO 100 IE = 1, Nedges
       HLLEW=FLUX_FUNC(Some arguments required)
       EDG_WK(IE)=HLLEW
100 CONTINUE
    DO 110 N = 1, Nnode
       NEDGE = IEDGE_in_NODE(0,N)
       DO 111 IE=1,NEDGE
          IEDGE=IEDGE_in_NODE(IE,N)
          XSIGN=SIGN(1.0D0,IEDGE)
          IEDGE=ABS(IEDGE)
          FLUX(N)=FLUX(N) + XSIGN*EDG_WK(IEDGE)
111    CONTINUE
110 CONTINUE
```

4.　計算性能測定条件

4.1.　ハードウエア

表 4.1 に、性能測定に使用した大規模クラスタ SMP システム CeNSS のハードウエア諸元について示す。

表 4.1　CeNSS ハードウエア諸元

ハードウエア	Fujitsu PRIMEPOWER HPC2500
論理ピーク性能	9.3TFLOPS, 5.2GFLOPS/CPU
メモリ容量	3.6TBytes, 64GBytes/node
ノード数	56
C P U 数	1,792, 32/node
ノード内アーキテクチャ	SMP
C P U	SPARC64V
L2 キャッシュ容量	2MBytes, on chip
インターコネクト形状	Full crossbar
インターコネクト性能	4GBytes × 2　(送受信)

4.2.　ソフトウエア

表 4.2 に使用したコンパイラ、リンケージエディタ及び MPI ライブラリのバージョン情報を示す。また、表 4.3 に翻訳時に指定したコンパイルオプションを示す。

表 4.2　ソフトウエアバージョン情報

ソフトウエア	バージョン情報
コンパイラ	Fujitsu Fortran Version 5.6
リンケージエディタ	Software Generation Utilities Solaris Link Editors: 5.8-1.296
MPI ライブラリ	MPI Patchlevel 2.21.0 MPLib version MPLIB-sr2.3.1

表 4.3　指定したコンパイルオプション一覧

コンパイルオプション
-Umpi -Qi,p -NRtrap -Kparallel -Kvppocl -Et

4.3.　JTAS 実行条件

4.3.1.　初期条件

性能測定のために設定した初期条件を表 4.4 に示す。

表 4.4　性能測定に使用した主な初期条件

設定データ	設定値	
時間積分ステップ数	500 ステップ	
クーラン数	1.0×10^{5}	
迎角	0.0 度	
レイノルズ数	1.0×10^{6}	
比熱比 γ	1.4	
自由流温度	273.15 K	
壁面温度	0.5	
自由流マッハ数	0.8	
流れの種別	層流	
空力係数計算	あり	
境界流入条件	初期密度	1.0
	流入速度(x 方向)	9.5×10^{-1}
	初期圧力	1.0
	初期渦動粘性係数	20.0
	非スリップ境界	あり

4.3.2.　格子点数

性能測定のために設定した計算格子点の数を表 4.5 に示す。MPI による並列実行時には、分割された領域の間で情報の授受を行うために余分の格子点が付与される。ここでは、この情報授受のためにオーバーラップして設けられた格子点数が示されている。実際に解析が行われる格子点数は「正味」として示した。

表 4.5　性能測定に使用した計算格子点の数

要素名	要素			辺	格子点
	三角錐	三角柱	四角錐		
2 process					
proc.0	716,199	0	0	861,171	128,802
proc.1	481,662	141,636	879	870,779	160,656
小　計	1,197,861	141,636	879	1,731,950	289,458
合　計			1,340,376	1,731,950	289,458
正　味					
小　計	1,173,840	141,636	879	1,690,955	280,969
合　計			1,316,355	1,690,955	280,969

5.　計算性能測定結果及び評価

以下に、JTAS の性能測定結果及びその評価について示す。測定は、MPI による並列化におけるプロセス数を２で固定とし、スレッド並列については１プロセス当たりのスレッド数１、２、４及び８の４ケースについて行った。尚、測定は他のジョブも存在する通常の運用状態の元で行った。このため、特に経過時間の測定結果には変動要因が含まれていることに注意されたい。以下、先ず全体の測定結果について概観した後、主な処理ごとの測定結果について示し、簡単な評価を行う。

5.1.　全体の測定結果及び評価

表 5.1 及び図 5.1 に JTAS 全体及び時間積分計算に要した経過時間を測定した結果及びスレッド並列化により得られた加速率を示す。表 5.1 において、各欄の上段が秒を単位とした経過時間であり、下段が加速率

である。また、経過時間はバリア同期を取ってルートプロセス（ランク０）における測定結果のみを示している（以下同様）。

表 5.1 より、いずれのスレッド数による実行においても、オリジナルに比べてスレッド版における経過時間が短縮されており、最適化の効果が確認できた。また、８スレッド実行において、加速率が全体では６倍を下回っているものの、時間積分の計算においては６倍を上回っており、当初の目標を達成する十分な加速率が得られた。

表 5.2 及び図 5.2 にスレッド版のプロファイラによる実行状況の解析結果について、その抜粋を示す。表中、各解析項目の"Rank0"及び"Rank1"は、MPI プロセスのランク０及びランク１における解析結果である。また、"Average"は、ランク０とランク１の測定値を単純平均した値であり、"Total"はプロファイラが"Process Total"として出力した結果である。L2 キャッシュ・ミス率、アドレス変換バッファ・ミス率共に十分に小さいとは言えず、この結果、浮動小数点演算性能も 150MFLOPS を若干上回る程度に留まった。

表 5.3 及び図 5.3 に経過時間を元に算出した JTAS オリジナルに対してスレッド版でどの程度性能が向上したかを表す性能向上率を示す。性能向上率は、オリジナルの実行に要した経過時間をスレッド版の実行に要した経過時間で除したものとして計算した。いずれのスレッド数による実行でも性能が向上することが確認出来た。

表 5.1　JTAS 全体の経過時間測定結果

上段：経過時間［秒］
下段：加 速 率［－］

測定区間	1Thread	2Thread	4Thread	8Thread
オリジナル全体	8803.71	4978.60	2770.26	1814.81
	1.00	1.77	3.18	4.85
スレッド版全体	6012.07	3229.03	1810.94	1075.91
	1.00	1.86	3.32	5.59
オリジナル時間積分	8733.57	4908.60	2703.89	1748.62
	1.00	1.78	3.23	4.99
スレッド版時間積分	5889.41	3106.27	1688.14	954.76
	1.00	1.90	3.49	6.17

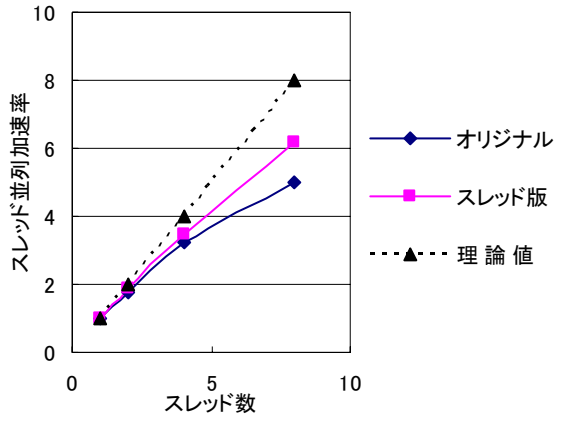
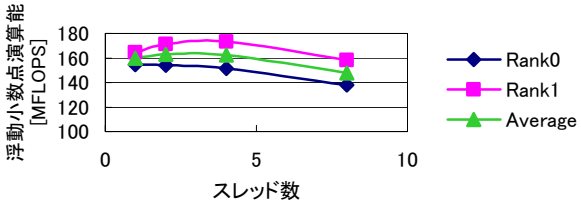


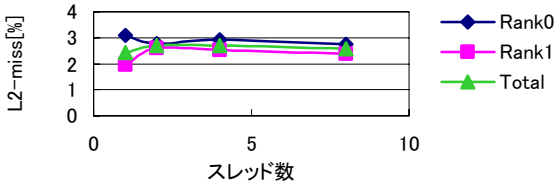
図 5.1　時間積分計算におけるスレッド並列実行加速率

表 5.2　スレッド版プロファイラ情報抜粋

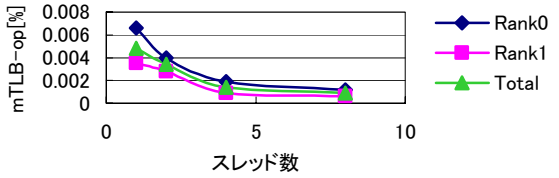
Rank	1Thread	2Thread	4Thread	8Thread
浮動小数点演算性能 FLOPS［MFLOPS］				
Rank0	154.8	154.0	151.4	137.7
Rank1	164.4	171.1	173.1	158.1
Average	159.6	162.6	162.3	147.9
L2 キャッシュ・ミス率 L2-miss［%］				
Rank0	3.08	2.78	2.91	2.75
Rank1	1.96	2.61	2.53	2.39
Total	2.42	2.69	2.71	2.57
アドレス変換バッファ・ミス率 mTLB-op［%］				
Rank0	0.0066	0.0040	0.0019	0.0012
Rank1	0.0035	0.0028	0.0009	0.0006
Total	0.0048	0.0034	0.0014	0.0009



(a) 浮動小数点演算性能



(b) L2キャッシュ・ミス率



(c) アドレス変換バッファ・ミス率

図 5.2　スレッド版プロファイラ情報抜粋

表 5.3　最適化による性能向上率

1Thread	2Thread	4Thread	8Thread
1.46	1.54	1.53	1.69

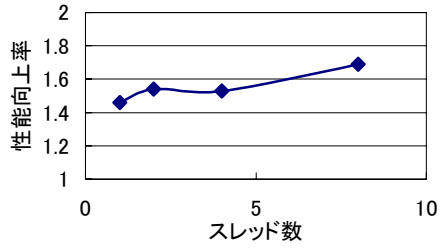


図 5.3　最適化による性能向上率

5.2. 流束の計算における測定結果及び評価

表 5.4 及び図 5.4 から 5.5 に、流束の計算について経過時間の測定を行なった結果を示す。

流束の計算においては、オリジナルに比べて処理に要する経過時間が短縮されると同時に、表 5.4 及び図 5.5 に示すように 8 スレッド（2 プロセス、16CPU）実行においてオリジナルで 5.55 倍だったスレッド並列による加速率がスレッド版において 7.42 倍に向上しており、最適化による効果が確認できた。

表 5.4　流束の計算における測定結果

上段：経過時間［秒］
下段：加 速 率［－］

version	1Thread	2Thread	4Thread	8Thread
オリジナル	1141.31	773.14	374.34	205.75
	1.00	1.48	3.04	5.55
スレッド版	1036.89	531.88	273.57	139.73
	1.00	1.95	3.79	7.42

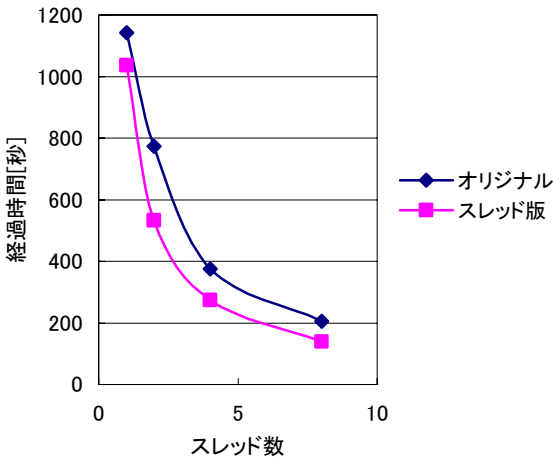


図 5.4　流束の計算における測定結果

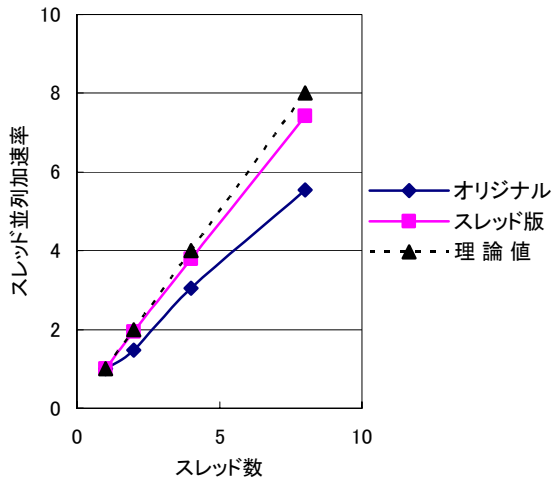


図 5.5　流束の計算におけるスレッド並列実行加速率

5.3. 勾配の計算における測定結果及び評価

表 5.5 及び図 5.6 から 5.7 に、勾配の計算について経過時間の測定を行なった結果を示す。

勾配の計算においては、8 スレッド実行の場合にオリジナルで 7.39 倍あったスレッド並列化による加速率がスレッド版では 7.00 倍に低下している（表 5.5 参照）。しかし、処理に要した経過時間は 362.70 秒から 237.36 秒に短縮されており最適化による計算性能向上の効果が確認された。

表 5.5　勾配の計算における測定結果

上段：経過時間［秒］
下段：加 速 率［－］

version	1Thread	2Thread	4Thread	8Thread
オリジナル	2679.61	1432.21	689.44	362.70
	1.00	1.87	3.89	7.39
スレッド版	1660.64	862.60	454.79	237.36
	1.00	1.93	3.65	7.00

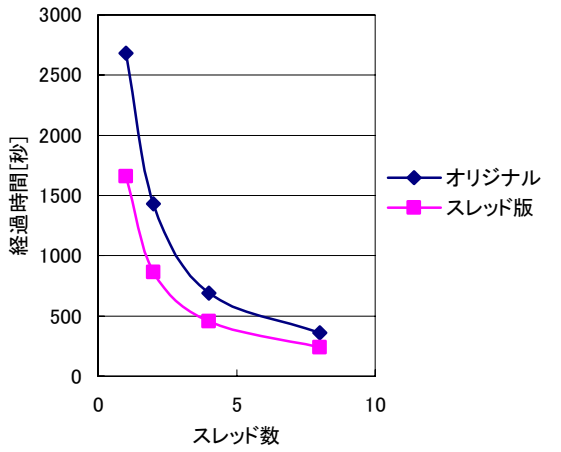


図 5.6　勾配の計算における測定結果

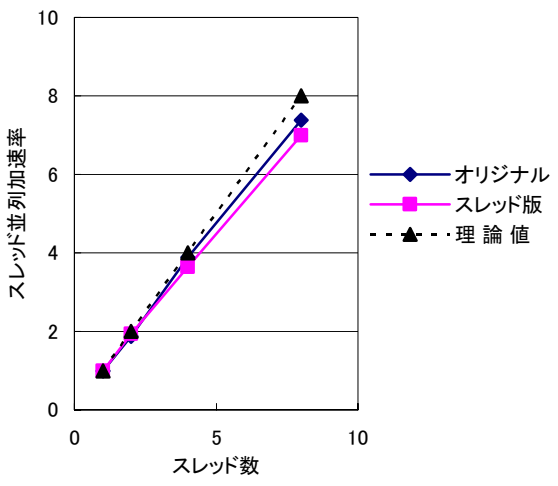


図 5.7　勾配の計算におけるスレッド並列実行加速率

5.4. 制限関数の計算における測定結果及び評価

表 5.6 及び図 5.8 から 5.9 に、制限関数の計算について経過時間の測定を行なった結果を示す。

制限関数の計算においては、8 スレッド実行の場合にオリジナルで 7.54 倍あったスレッド並列化による加速率がスレッド版では 6.71 倍に低下している。しかし、処理に要した経過時間は、211.96 秒から 161.96 秒に短縮されており最適化の計算性能向上の効果が確認された。

表 5.6 制限関数の計算における測定結果

上段：経過時間［秒］
下段：加 速 率［－］

version	1Thread	2Thread	4Thread	8Thread
オリジナル	1599.02	861.88	410.54	211.96
	1.00	1.86	3.89	7.54
スレッド版	1087.35	581.07	310.65	161.96
	1.00	1.87	3.50	6.71

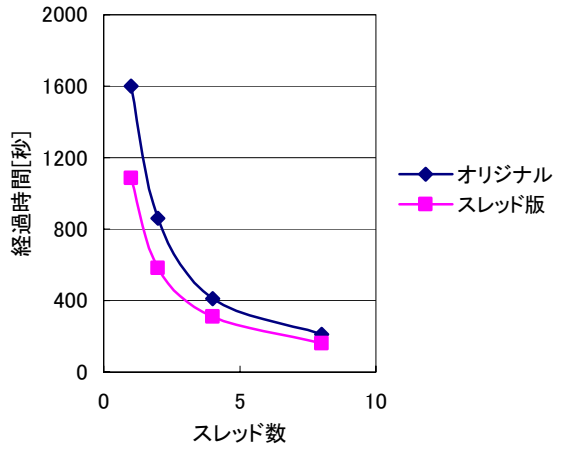


図 5.8 制限関数の計算における測定結果

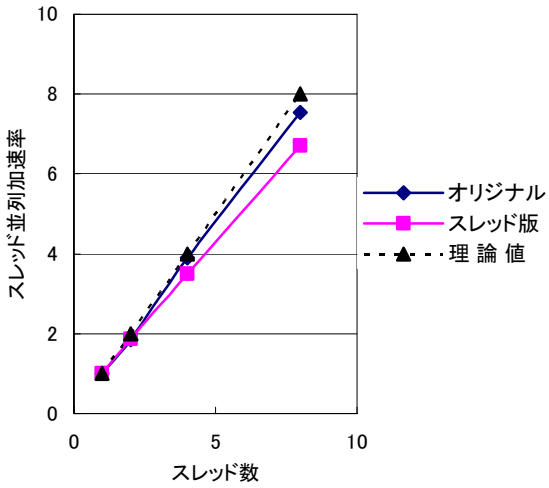


図 5.9 制限関数の計算におけるスレッド並列実行加速率

5.5. LU－SGS 法の計算における測定結果及び評価

表 5.7 及び図 5.10 から 5.11 に、LU－SGS 法の計算について経過時間の測定を行なった結果を示す。

LU－SGS 法の計算においては、8 スレッド実行の場合にオリジナルで 2.97 倍であったスレッド並列化による加速率がスレッド版では 5.76 倍に向上した。同時に、処理に要した経過時間も 773.07 秒から 238.55 秒に短縮されており最適化の効果が確認された。

表 5.8 及び図 5.12 にコーディング方法を変更した場合に、LU－SGS 法の計算に要する経過時間がどのように変化するかを示す。コーディング方法としては、(1) 色分けにより再帰参照を回避した場合（オリジナル、リスト 2.2 参照、「色分け」と表記）、(2) 色分けを削除し DO ループを分割した場合（リスト 3.1 参照、「色分け削除」と表記）、(3) 色分けによる再帰参照の回避を行うとともに、超平面を考慮した節点番号の付け替え（Reordering）を行った場合（スレッド版、「色分け+Reo.」と表記）、(4) 色分けを削除するとともに DO ループの分割を行い、さらに、節点番号の付け替えを行った場合（「色除+Reo.」と表記）の四種類を検討した。これより明らかなように、色分けの削除を行うとオリジナル（色分けによる再帰参照の回避）に比べて性能が低下すること、節点番号の付け替えを行った場合に最も良い性能が得られることが確認できた。色分けの削除を行った場合について LU－SGS 法の計算に含まれるあるサブルーチンについて調べてみると、MPI 並列のみによる実行の場合にリスト 3.1 DO 100 に相当する部分の計算時間に対して、単に節点毎の配列に足し込むのみである DO 110 に相当する部分の計算に 60%以上の時間を要しており、このことが色分けを削除し DO ループを分割した場合に性能が低下する原因であると考えられる[14]。以上のことから、スレッド版では節点番号の付け替えのみを行い色分けの削除は行わなかった。

表 5.7 LU－SGS 法の計算における測定結果

上段：経過時間［秒］
下段：加 速 率［－］

version	1Thread	2Thread	4Thread	8Thread
オリジナル	2297.07	1414.62	964.67	773.07
	1.00	1.62	2.38	2.97
スレッド版	1374.72	699.87	389.80	238.55
	1.00	1.96	3.53	5.76

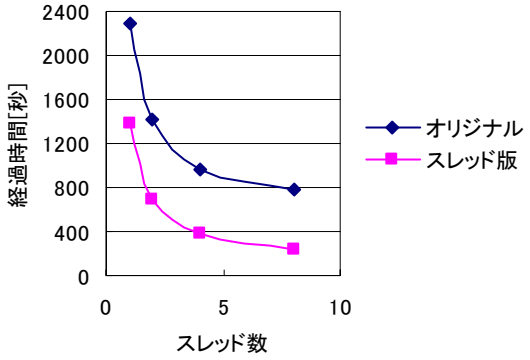


図 5.10　LU－SGS 法の計算における測定結果

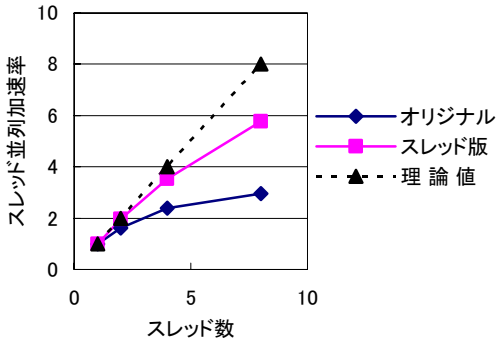


図 5.11　LU－SGS 法の計算におけるスレッド並列実行加速率

表 5.8　LU－SGSの計算におけるコーディング方法と性能の関係

経過時間［秒］

version	01Thread	02Thread	04Thread	08Thread
(1) 色　分　け　（オリジナル）	2297.07	1414.62	964.67	773.07
(2) 色分け削除	2671.32	1950.94	1638.87	1269.93
(3) 色分け＋Reo.（スレッド版）	1374.72	699.87	389.80	238.55
(4) 色　除＋Reo.	1789.36	896.81	485.75	278.96

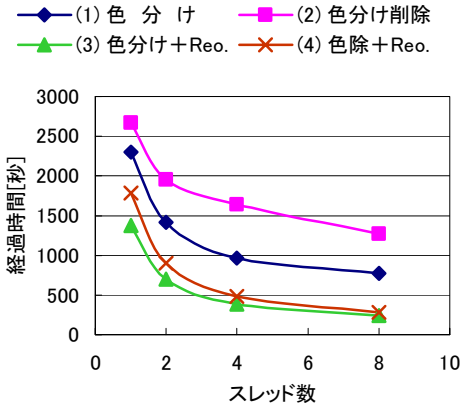


図 5.12　LU－SGS法の計算における
コーディング方法と性能の関係

5.6.　スレッド並列化におけるオーバーヘッド

勾配の計算及び制限関数の計算において、オリジナルに比べてスレッド版の計算性能は向上したものの、スレッド並列化による加速率は低下した。

そこで、加速率 S 、並列化率 α 及びスレッド数 n としてアムダールの法則、

$$S=\frac{1}{\frac{\alpha}{n}+(1-\alpha)} \qquad (6.1)$$

を適用し、スレッド並列化によるオーバーヘッド O を

$$O=(1-\alpha)T \qquad (6.2)$$

によって評価してみる。但し、 T は当該測定区間の計算に要した経過時間であり、従って、オーバーヘッドは、当該区間の計算に要する時間の内、並列化されていない部分の処理に要した時間である。この時、勾配の計算において、オリジナルで 98.8%であった並列化率 α がスレッド版では 98.0%に低下し、オーバーヘッドは、オリジナル版の 4.2 秒に対してスレッド版では 4.9 秒に増加している。この性能低下の原因は、リスト 3.1 の DO 110 に相当する計算のオーバーヘッドが大きいためである[14]。とはいえ、計算時間はオリジナル版に比べて短縮されておりチューニングとしての効果はあったと言える。

6.　まとめ

3 次元ハイブリッド非構造格子有限体積法 Euler／Navier-Stokes ソルバ JTAS （JAXA Tohoku university Aerodynamic Simulation code）について、全体性能の向上とスレッド並列化の最適化を目的とした変更を加え、JTAS スレッド並列版を開発して、全体の性能が約 1.5 倍向上し、時間積分計算部分で、8 スレッド実行によるスレッド並列化加速率が約 6.2 倍と、理論値の 7 割を越える性能が得られることを確認した。

参考文献

[1] Iwamiya, T., "NAL SST Project and Aerodynamic Design of Experimental Aircraft", Proceedings of the 4th ECCOMAS Computational Fluid Dynamics Conference, Wiley, Chichester, England, U.K., pp. 580-585, 1998.

[2] 高木正平, 坂田公夫 他.,“［特集］超音速実験機計画について”, 日本流体力学会誌ながれ 18-5, pp.275-307, 1999.

[3] 藤田健, 松島紀左, 中橋和博., "非構造格子 CFD を用いた逆問題設計システムの高度化", 第 15 回数値 流体力学シンポジウム予稿集 D05-3, 2001.

[4] 高橋克倫, 藤田健 他., "NAL 小型超音速実験機 NEXST-1 の結合分離金具形状修正の CFD 解析", 第 17 回数値流体力学シンポジウム予稿集 F2-3, 2003.

[5] "小型超音速実験機（NEXST-1）の舵角変化時における空力特性変化の数値解析", http://www.ista.jaxa.jp/res/c02/a06_01.html

[6] Nakahashi, K., Ito, Y., and Togashi, F, “Some challenges of realistic flow simulations by unstructured grid CFD”, Int. J. for Numerical Methods in Fluids, Vol.43, pp.769-783, 2003.

[7] "並列型の非構造格子ソルバープログラム　ユーザーズマニュアル"　平成 15 年 2 月 28 日　財団法人青葉工業振興会.

[8] Obayashi, S. and Guruswamy, G. P., "Convergence Acceleration of a Navier-Stokes Solver for Efficient Static Aeroelastic Computation", AIAA Journal, Vol. 33, No. 6, pp.1134-1141, 1995.

[9] Venkatakrishnan V., "On the Accuracy of Limiters and Convergence to Steady State Solutions.", AIAA Paper, 93-0880, 1993.

[10] Sharov, D. and Nakahashi, K., "Reordering of 3-D Hybrid Unstructured Grids for Vectorized LU-SGS Navier-Stokes Computations", AIAA 97-2102, 1997.

[11] Sharov, D. and Nakahashi, K., "Reordering of 3-D Hybrid Unstructured Grids for Lower-Upper Symmetric Gauss-Seidel Computations", AIAA Journal, Vol. 36, No. 3, 1998

[12] Sharov, D. , Luo, H. and Baum. J. D., "Implementation of Unstructured Grid GMRES+LU-SGS Method on Shared-Memory, Cache-based parallel Computers.", AIAA 2000-0927, 2000.

[13] Fujita, T, et. al, "Evaluation of Parallelized Unstructured-grid CFD for Aircraft Applications", Proc. of Parallel CFD 2002.

[14] 坂下雅秀,松尾裕一,村山光宏., 「非構造格子 Euler/Navier-Stokes ソルバ JTAS の計算性能最適化」, 宇宙航空研究開発機構研究開発報告, 2006, 投稿中.