低通信超並列チャンネル流 DNS コードの開発

山本義暢

山梨大学大学院総合研究部

１．はじめに

壁面に沿って流れる乱流場（壁乱流）は、流体物理学上はもちろん、工学的にも重要であり、乱流理論・モデリング・制御の検証に最も多く適用される流動場である。本研究ではこの壁乱流のカノニカル流であるチャンネル流(図 1 参照)を対象とし、世界最高レイノルズ数(Re)条件(表 1 参照)を実現する世界最速直接数値計算(Direct Numerical Simulation, DNS)コードを開発している。

乱流の DNS における空間離散化手法としては周期境界条件が適用できる場合、フーリエ・スペクトル法が最も高精度かつ効率的であり、標準手法として確立している。しかしフーリエ・スペクトル法の主要演算部となる高速フーリエ変換(Fast Fourier Transform, FFT)は並列性がなく、並列計算においては分割軸の転置により並列性を作る必要が生じる。この転置は一般に all-to-all 通信(a2a)となり近年の疎結合超並列計算機が最も苦手とする通信タイプとなる。そのためスペクトル法を用いた大規模乱流 DNS の実現には、通信量を最低とする並列化手法及びノード配置の検討が不可欠である。

本報告では、大規模 HPC チャレンジを利用して実施した wisteria BDEC-01(Odyssey) 6144 ノード上における通信量を最低とする最適ノード配置の検討とその性能評価結果を紹介する。

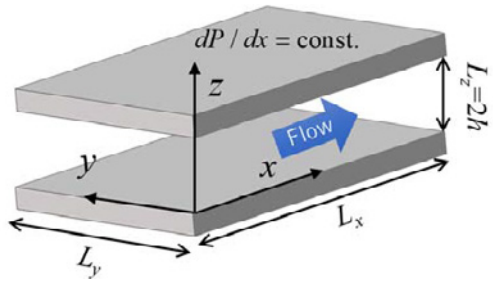


図 1　チャンネル流の体系と座標系

表 1　計算条件

Re_τ	L_x /h	L_y /h	N_x Δx^+	N_y Δy^+	N_z Δz^+
16000	16	6.4	14400 17.8	13824 7.4	5760 0.6-12.0

$Re_\tau = u_\tau h/\nu$: friction Reynolds number. u_τ: friction velocity, h: channel half width, ν: kinematic viscosity, L_x: streamwise computational length, L_y : spanwise computational length, N_x (Δx), N_y (Δy), N_z (Δz,): grid number (resolution) for stream (x), spanwise (y), and wall-normal (z) directions, respectively.

２．計算対象と領域分割及び並列化手法

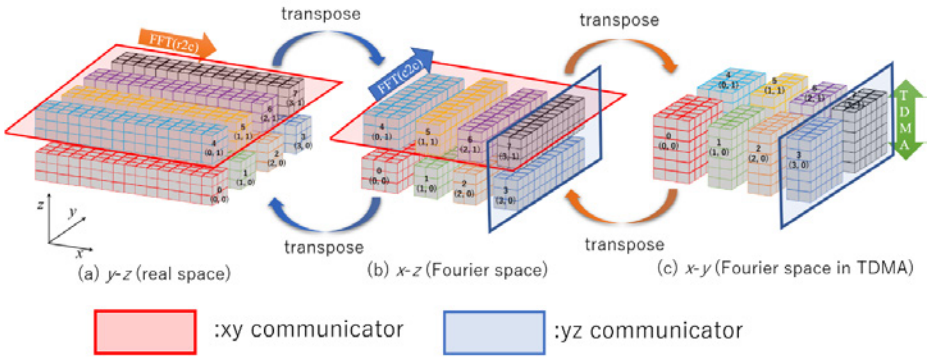
計算対象は図 1 に示す 2 枚の平板間を一定の圧力勾配により流れる十分に発達したチャンネル流である。流れ場の支配方程式は非圧縮性流体の Navier-Stokes 式と連続式であり、主流(x)及びスパン方向(y)には速度及び圧力に周期境界条件、壁面(z=0, 2h)では速度場に no-slip 条件、圧力場に対称条件が適用できる。空間離散化手法として、x, y 方向フーリエ・スペクトル法、z 方向に 2 次精度中心差分法を適用する。従って本 DNS コードの主要演算は、x, y 方向への 2 次元 FFT と z 方向への 3 重対角行列解法(Tri-Diagonal Matrix Algorithm, TDMA)となる。

並列化においては世界最大 Re 数規模を対象とするために、数百万並列に対応すべく、y(MPI におけるコミュニケータ：mpi_xy_world)及び z(同：mpi_yz_world)方向への 2 次元領域分割を行う。一般には図 2(a)に示すような 4 段階の転置が発生する [1], [2]。この場合、フーリエ正変換過

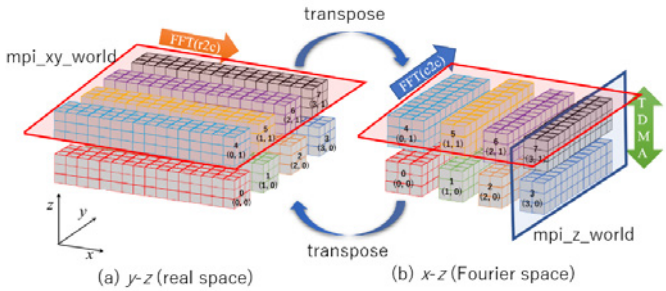
程における x 方向の FFT においては完全な並列性が得られるが、y 方向の FFT において並列性を確保するために分割軸を x に変更する($y \rightarrow x$)。さらに z 方向の TDMA において並列性を確保するために分割軸を y に変更する($z \rightarrow y$)。フーリエ逆変換時はこの逆の過程にて実施し、正・逆過程において 4 段階転置が必要となる。この転置は各コミュニケータ内での a2a となり通信負荷は極めて高く、乱流 DNS における大規模化(高レイノルズ数への対応)において大きな障壁となっている。

そこで本 DNS コードにおいては TDMA において転置を伴わない並列化手法(隣接シフト通信による逐次計算と x, y 方向へのオーバラップ化)を開発し [3], [4]、図 2(b)のように 2 段階の転置に抑え、通信負荷を低減させている。一方 y 方向の FFT においては、分割軸の転置($y \rightarrow x$)により並列性を確保する。従ってこの転置(a2a)は依然として残るが、mpi_xy_world を物理ノード上における直接(あるいは近接)接続ノードに割り当てることにより、a2a 負荷を低減させる。FX システムにおいては 1 or 2 tofu 座標(1tofu 座標＝12 ノード)の近接結合の 12 ノード程度を mpi_xy_world に割り当て、領域分割 ＝ ノード形状 ＝ $12 \times N$ (N は z 方向のノード数)とすると高効率が期待できる。表 1 の計算条件においてはメモリ量で約 148TB を要するため、ノードあたりのメモリ量約 28GB の FX1000 では 5760 ノードが最低実行条件であり、この場合 $N = 480$ となる。

また本 DNS コードでは、OpenMP 機能を用いた FFT 演算と a2a のオーバラップ化を行っており、FFT 演算の約 90%以上を a2a 実行中に同時実行し、通信時間の隠ぺいを図っている。



(a) 従来手法:4 段階転置



(b) 本手法:2 段階転置

図 2 2 次元領域分割方法、4×2(=8 並列)の場合

３．Tofu インタコネクト D とその問題点

Wisteria-O (Odyssey)のインタコネクトは Tofu インターコネクト D(6 次元メッシュ/トーラス)であり、(a, b, c) = (2, 3, 2) の 12 ノードを 1tofu 座標とし、この tofu 座標が 3 次元 (X, Y, Z) = (10, 8, 8)メッシュにより構成されている。2D ノード形状指定時は (X*a*b, Y*Z*c), 3D ノード形状時は (X*a, Y*b, Z*c)のノードを使用することになる。今回の大規模 HPC チャレンジでは、(X, Y, Z) = (8, 8, 8) が利用可能なため、最大構成は、6144 ノード = 48×128 (2D ノード形状指定時) = 16 × 24 × 16 (3D ノード形状指定時)となる。本 DNS コードでは上述の通り、領域分割として mpi_xy_world に 12 ノードを割り当てることにより a2a 負荷を低減させる。2D ノード形状指定時は、12 ×128 (=1536)ノードが最大ノード数となり、全体の 1/4 のノードに抑えられてしまう。もちろん mpi_xy_world に 48 ノードを割り当てれば、6144 ノードが利用できるが、FFT 担当ノードが 48 ノードとなるため a2a コストが跳ね上がる。つまり 2D 領域分割 ＝ 2D ノード形状 として高効率を維持するためには、FX システムにおいては全体の 1/4 程度のノードしか利用できない問題が生じる。

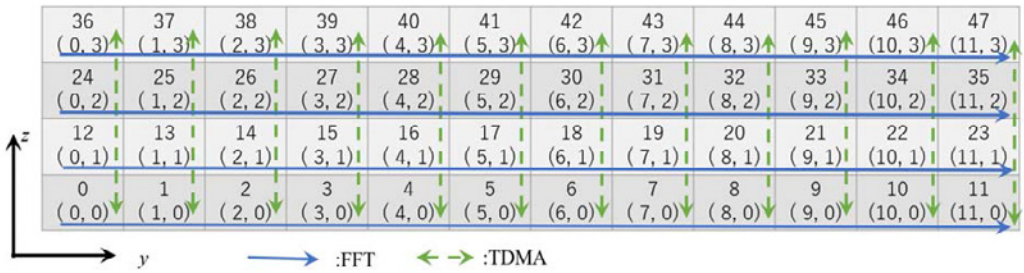
４．3D ノード形状時における 2 次元領域分割の最適配置の検討

本 DNS コードにおいて FX システムの最大構成近くを利用しかつ、高効率を維持するには、ノード形状 3D 上において、2D 領域分割を通信量が最低となるように配置する工夫が必要となる。このためまず 3D ノード形状時における tofu 座標配置について分析を行った。

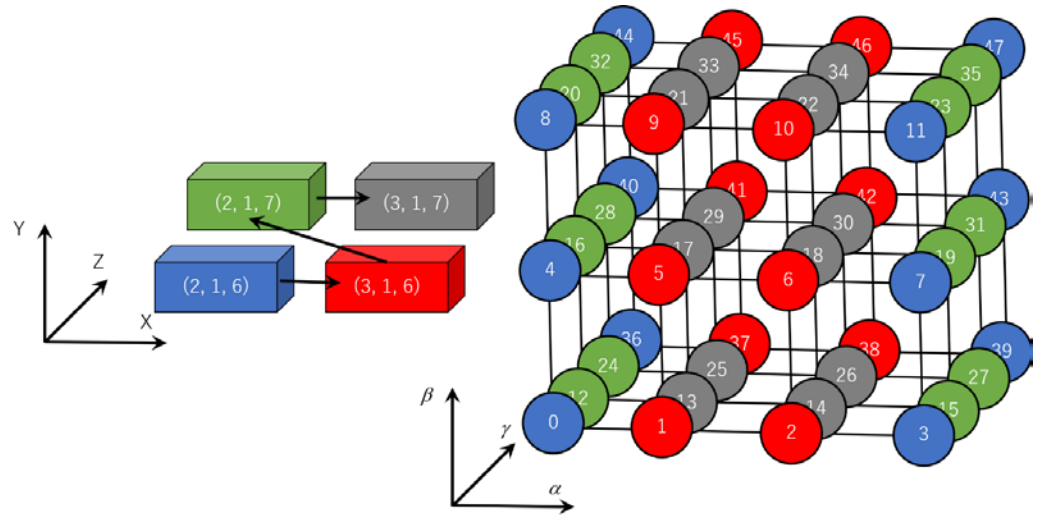
図 3 は 3D のノード配置:α × β ×γ = 4 × 3× 4 (= 48 ノード:torus)を指定し、2 次元領域分割: 12 × 3 の場合におけるランク及びノード配置と tofu 座標(1tofu 座標は 12 ノードから構成)の関係を示す。またここでは簡単のため 1MPI プロセス/1 ノードとして標記している。図 3(a)はチャンネル流の (y, z) 方向への 2 次元領域分割を示し、数字は mpi_comm_world のランク番号(mpi_xy_world のランク番号、mpi_z_world のランク番号)を意味する。図 3(b)左は 48 ノードの tofu 座標 (X, Y, Z) 配置を示す。48 ノードの場合、(X, Z) 面内の直接結合された 4tofu 座標から構成されてる。一方図 3(b)右は 3D のノード配置 (α, β, γ) を示し、数字は mpi_comm_world のランク番号、色はどの tofu 座標にあるか、を示している。このノード配置(デフォルトの 3D ノード形状)の場合、FFT は 2 tofu 座標間で行われることがわかる。一方、TDMA は隣接シフト通信のため tofu 座標間で連続していれば、1 ホップ(中継なし)に通信を行うことができる。図 3(b)左の場合は、**(2, 1, 6)** → **(3, 1, 6)** → **(2, 1, 7)** → **(3, 1, 7)** の順にノードが割り当てられるが、**(3, 1, 6)** → **(2, 1, 7)**間は tofu 座標上で隣接していないため 2 ホップの通信が生じることがわかる。

そこで図 3(c)左のように tofu 座標間で連続となるように **(2, 1, 6)** → **(3, 1, 6)** → **(3, 1, 7)** → **(2, 1, 7)** の順にノードを割り振ると隣接シフト通信のホップ数は 1(直接通信)により実行できる。さらに図 3(c)右のように 3D ノード上のランク配置を MAP により指定すれば、FFT に a2a は 1tofu 座標内の通信で行うことができる。

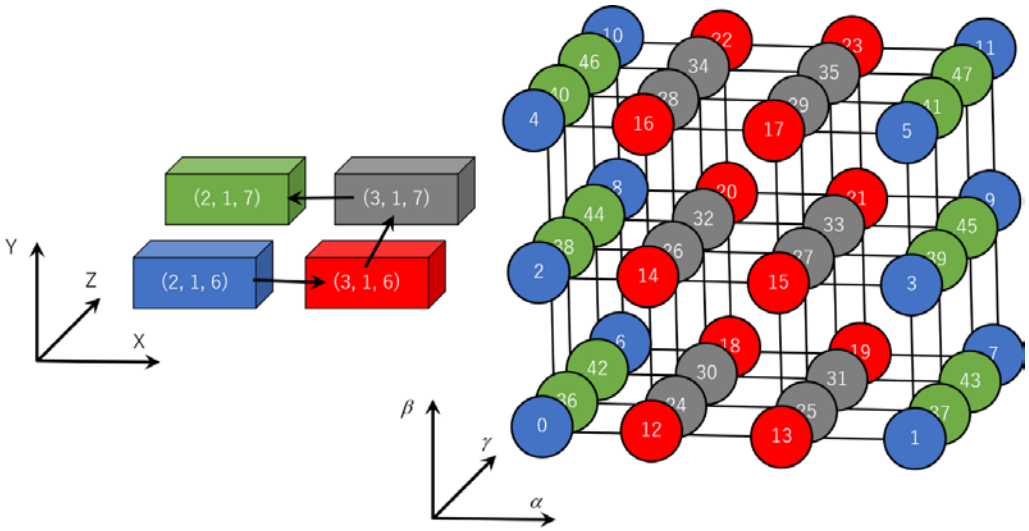
この MAP の作成手順としては、ダミープログラムにて 3D ノード形状を strict 指定により実施し、図 3(b)の tofu 座標と 3D ノード配置との情報を取得する(stats 情報を出力させるとこの情報が得られる)。この情報により図 3(c)のように FFT 担当の 12 ノード単位を 1tofu 座標に割り当て、かつ tofu 座標上で連続となるようにランクを割り振ることになる。数千から数万ノードを対象とする場合、手作業でこれを行うことは困難なため、一連の過程を自動化するコードを作成した。



(a) 2 次元領域分割: 12 × 3 の場合



(b) 3D ノード形状時の tofu 座標とランク番号の関係(デフォルト時) : 4 × 3× 4



(c) 3D ノード形状時の tofu 座標とランク番号の関係(最適化時) : 4 × 3× 4

図 3　3D ノード形状において 2D 領域分割における通信量を最低とするノード配置(48 ノードの例)

５．世界最高レイノルズ数条件でのベンチマーク

4.で示した 3D ノード形状上で 2 次元領域分割時の通信量を最低とするノード配置最適化手法を Wisteria-O (Odyssey) 6144 ノード(294,912 コア)に適用した。計算条件は表 1(世界最高レイノルズ数条件)とし 100step の時間積分により計測を行った。ただし実際の乱流計算では乱流場(初期条件)を読み取る必要があるが、本ベンチマークでは疑似乱流場を作成し、データ読み取りは行っていない。ノード配置条件としては、(1) 2D ノード形状指定＝2D 領域分割、(2) 3D ノード形状上で 2D 領域分割をデフォルト配置(図 3(b)相当)、(3)3D ノード形状上で 2D 領域分割を最適化した場合(図 3(c)相当)、の 3 ケースを実行した。またノード内は全ケースとも 12MPI × 3AP(OpenMP & 自動並列)を適用した。

計測結果を表 2 に示す。RUN(1)のノード形状=領域分割の場合は、3.で示したように FFT 担当ノードが 48 ノードとなる。この時 FFT 担当の 48 ノードは 8 tofu 座標で配置されていた。これはトーラス指定に対応するためと思われる(従って本 DNS コードでは 2D ノード形状時のトーラス指定を外した方がいいかもしれない)。この 8 tofu 座標間の FFT 及び a2a により、1 ステップあたり約 10 秒を要する。RUN(2)の 3D ノード形状時は、FFT 担当 12 ノードが 8 あるいは 12 tofu 座標に配置されている。このため a2a コストが(1)よりも増加し、1 ステップあたり約 13.3 秒を要した。RUN(3)の 3D 形状時に最適ノード配置を用いた場合は、FFT 担当 12 ノードが 1tofu 座標内に配置され、かつ tofu 座標間で連続となる。そのため 1 ステップあたり 4.63 秒と RUN(1)の 2.3 倍高速となった。

表 2　世界最高レイノルズ数条件での計測結果

RUN	ノード数	ノード形状	領域分割	s/step	TFLOPS	備考
(1)	6144	48×128	48×128	10.73	166	ノード形状=領域分割
(2)	6144	16×24×16	12×512	13.30	134	デフォルト
(3)	**6144**	**16×24×16**	**12×512**	**4.63**	**385**	**最適 MAP 使用**

６．理化学研究所「富岳」への応用

本研究の 3D ノード配置における 2D 領域分割の最適ノード配置方法は、同様のインタコネクトをもつ理化学研究所「富岳」へも適用可能である。「富岳」の large クラスは、2D ノード形状は 144 × 816 (但し、利用ノード数 12288 以下)、12288 ノードを超える middle クラスは 144 × 576(但し、利用ノード数 55296 以下)となっている。従って、表 1 の条件においては、2D ノード形状 = 領域分割の設定(12× N)においては、N = 720 の 8640 ノード程度が上限となる。これを超えるノード数を利用するには、24× N(N=480, 576) のように FFT 担当ノードを拡張する必要があった。図 4 は富岳での実行結果であり、ノード形状＝領域分割とし、□が 12× N (N=480, 720)、▲が 24× N(N=480,576) の結果である。12× N の場合はほぼ 100%の並列化効率が得られているが、これ以上ノード数を増やすことができず、FFT 担当ノード数を 24 とすると性能が劣化することがわかる。一方本研究で開発した 3D ノード形状で 2D 領域分割を最適配置する方法(■)においては、領域分割：12×960(=11520 ノード), 12×1440(=17280)ノードにおいて 12× N の場合(□)とほぼ同等の性能でノード数を増やすことが可能になっている。17280 ノードを用いた場合の演算速度は約 1050 TFLOPS であり、チャンネル流 DNS における最速値となっている(従来の最速値は Lee et al [2], による IBM Mira 全系を利用した 271 TFLOPS)。

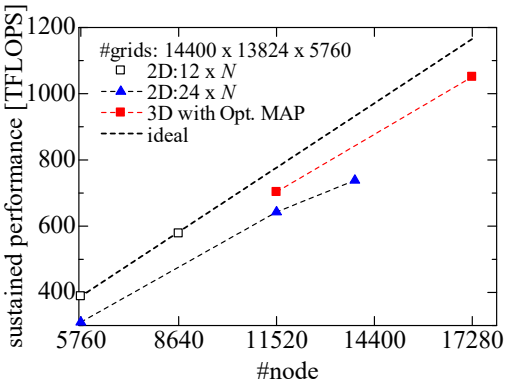


図 4　本手法の「富岳」への応用：表 1 の実行条件での検証、■が本手法の結果

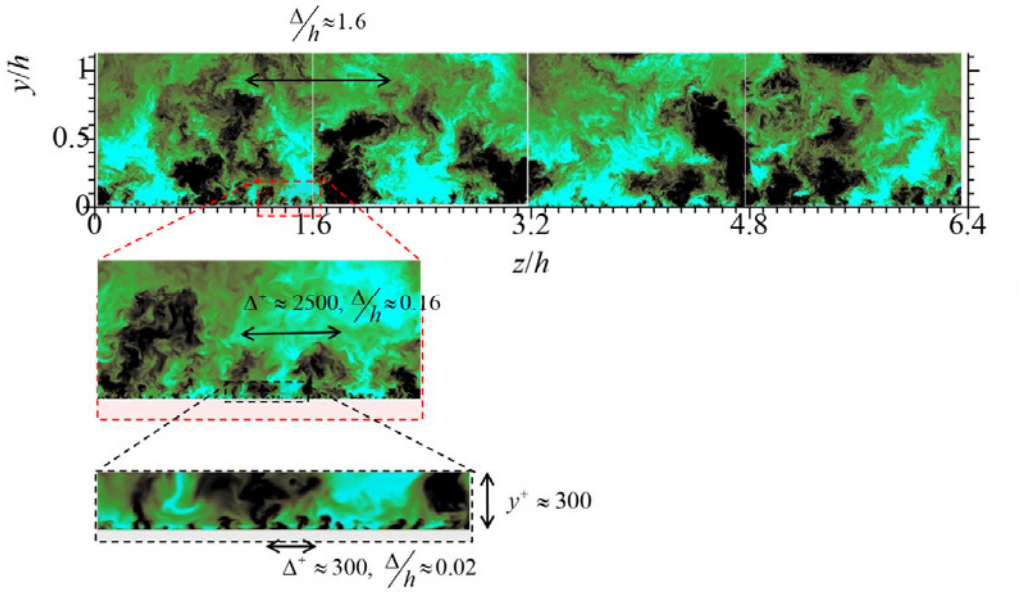


図 5 高レイノルズ数壁面乱流場における階層構造：主流方向速度乱れ(u)のコンタープロット、-3(black) < u^+ < 3(green)、endview (チャンネルの下半分:0 < y/h < 1), なお本可視化においては y 軸：壁垂直方向、z 軸：スパン方向としている。

7．世界最高レイノルズ数条件での PRODUCTION RUN

表 2 の RUN(1)及び(3)の条件で、PRODUCTION RUN (初期乱流場を読み取り、最後にリスタート用に最終状態の乱流場を出力)を実施した。本条件では、約 33TB のデータ読み取りと書き込みをそれぞれ行う必要があり、実際の PRODUCTION RUN においては計算速度に加えて IO 速度も極めて重要となる。本 DNS コードでは、(x, y)平面内の代表ランクが IO を担当するため、RUN(1)では 128 ×4(速度 3 成分+圧力)並列により IO を実施する。同様に RUN(3)では 512×4 並列での IO となり並列数は RUN(3)が 4 倍多くなる。

実測した IO 速度は、RUN(1)が読み取り・書き込みのそれぞれに約 10 分に対し、RUN(3)が約 4 分(約 150 GB/s の IO 速度)と 2.5 倍程度高速であった。一方理化学研究所の「富岳」においては、並列数に依存せず 10-20 分とムラが大きく、かつ遅い IO であり、Wisteria-o におけるファイルシ

ステムの優位性が確認できた。この PRODUCTION RUN を 1 回あたり約 4 時間の実行時間で 5 回実施したがハード障害等の発生もなく安定した計算を行うことができた。

図 5 に PRODUCTION RUN による計算された瞬時乱流場の可視化を示す(なおここでは本分野の慣例に従い、壁垂直方向を y 軸、スパン方向を z 軸と置き換えている)。高レイノルズ数の壁面乱流場では階層的な縦渦構造が出現することが知られているが、本条件では、外層規模(~1.6h)、内層規模(~0.16h)、そして wall-unit(摩擦速度と動粘性により無次元化した単位)でスケールする特徴的な階層構造が存在することがわかる。なお本条件の速度 1 成分は約 8TB(単精度に落としても 4TB)の容量を要するため、全計算領域の可視化は実現できておらず今後の課題である。

８．まとめと今後の展望

スーパーコンピュータ「富岳」は京に比べ、ノードあたりの演算性能で 26 倍の向上を実現しているが、通信性能は 2 倍程度の向上にとどまっており、通信負荷の高い大規模並列コードの適用においては、その対策が不可欠となる。本 DNS コードでは all-to-all 通信を伴う転置を 2 段階に抑えた 2 次元領域分割方法、通信と演算とのオーバラップによる通信時間の隠ぺい、かつノード形状と領域分割を一致させ、FFT 担当ノードを 12 ノード程度に抑え all-to-all 通信負荷を低減させる手法を開発することにより高効率演算を実現している。しかし FX システム特有の tofu インターコネクトにおいて全系規模を対象とする場合には、2 次元ノード形状時の FFT 担当ノードが大幅に増加し、性能劣化をもたらす問題が生じていた。

そこで本研究では、3D ノード形状において 2 次元領域分割時の通信量を最低とするノード配置最適化手法を開発し、Wisteria-O (Odyssey) の全系規模 6144 ノード(294,912 コア)に適用しその性能評価を実施した。その結果、従来の 2D ノード形状＝2D 領域分割の場合に比べ、2.3 倍の高速化を達成し、384TFLOPS の演算速度(理論性能の約 1.8%に相当)を得た。本手法は Wisteria-O (Odyssey)のみならず、tofu インタコネクトによるすべての FX システムに適用可能である。つまり富岳全系規模への拡張も可能であり、本 DNS コードの適用性を大きく進展させるとともに、世界最大レイノルズ数壁面乱流場の DNS データベース構築の実現性を飛躍的に向上させるものである。

謝辞

本研究は Wisteria/BDEC-01 スーパーコンピュータシステムにおける「大規模 HPC チャレンジ」制度を利用して実施した。また科学研究費 JP26400410 及び公益財団法人柏森情報科学振興財団の助成を受けた。記して謝意を表する。

参 考 文 献

[1] J. Xu, "Benchmarks on tera-scalable models for DNS of turbulent channel flow," *Parallel Computing,* vol. 33, no. 12, pp. 780-794, 2007.

[2] M. Lee, N. Malaya and R. D. Moser, "Petascale direct numerical simulation of turbulent channel flow on up to 786k cores," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2013.

[3] Y. Yamamoto and Y. Tsuji, "Numerical evidence of logarithmic regions in channel flow at Reτ= 8000,"

Phys. Rev. Fluids, vol. 3, no. 1, p. 012602, 2018.

[4] Y. Kaneda and Y. Yamamoto, "Velocity gradient statistics in turbulent shear flow : an extension of Kolmogorov's local equilibrium," *J. Fluid Mech.,* vol. 929, p. A13, 2021.