平成 25 年度先端的計算科学研究プロジェクト

流体型宇宙プラズマシミュレーションコードの性能チューニング

梅田隆行（名古屋大学太陽地球環境研究所）
深沢圭一郎（九州大学情報基盤研究開発センター）

1. 研究の目的と意義

太陽から地球に至るジオスペース環境の変動を理解することは、人類の活動が宇宙へと拡大しつつある今日、極めて重要な課題である。人類の活動に影響を与えるジオスペースの変動現象としては、突発的な磁気嵐やオーロラの爆発現象、放射線高エネルギー粒子生成、高エネルギー粒子線による人工衛星の誤作動などが挙げられる。これらの現象は、電磁気圏プラズマのグローバルな対流循環、境界層で生起するメソスケールでの突発的な不安定性（平衡状態の破れ）及び、電子・イオンが粒子として振舞うミクロスケール現象（粒子加速・加熱）が複雑に結びついており、マルチスケール結合過程であり、宇宙天気と呼ぶ。本研究の大きな目的は、ジオスペースで生起する非線形プラズマ現象を解明し、宇宙環境変動の因果関係を理解すると共に、数値宇宙天気予報に適用することである。

多様な時空間スケールの非線形プラズマ現象を解明するために、グローバル現象を扱う磁気流体力学（MHD）／多流体モデル、ミクロ現象を扱う運動論（粒子／ブラソフ）モデル及び、両者の中間（メソ）スケール現象を扱う流体と運動論のハイブリッドモデルなどの様々なコードが発達してきた。本研究ではその中でも特に、流体型のシミュレーションコードであり、かつ（現行の MHD・粒子モデル、次世代の多流体・ハイブリッドモデルに対して）次々世代の宇宙プラズマシミュレーション手法であるブラソフモデルに着目しており、最新のスーパーコンピュータの能力を最大限に活用できるように、ブラソフコードの並列化・最適化を行うことを目的とする。その意義は、メソ・マクロスケールから粒子運動論的なミクロスケールまでをシームレスに扱う大規模シミュレーションにより、無衝突宇宙プラズマのマルチスケール結合過程の解明を目指すことにある。

本研究課題は、平成 24 年度にも先端的計算科学研究プロジェクトのベンチマーク課題として採択されており[1]、得られた結論として、FX10 と CX400 では最適なコーディングが異なることが挙げられる。また残された課題として、CX400 において 1024 ノード以上の測定ができなかったことが挙げられる。平成 25 年度は、CX400 において 1024 ノード以上で測定に失敗した原因を明らかにし、全ノード測定に再挑戦することが目的である。

2. ブラソフコードの概要

ブラソフコードは、無衝突宇宙プラズマの運動論シミュレーション手法の 1 つであり、位置－速度位相空間に定義されたプラズマ粒子の分布関数の時間発展を、以下のブラソフ（無衝突ボルツマン）方程式により直接解き進めている。

$$\frac{\partial f_s}{\partial t} + \vec{v}\frac{\partial f_s}{\partial \vec{r}} + \frac{q_s}{m_s}(\vec{E} + \vec{v} \times \vec{B})\frac{\partial f_s}{\partial \vec{v}} = 0 \tag{1}$$

ここで $\vec{E}$、$\vec{B}$、$\vec{r}$ 及び $\vec{v}$ はそれぞれ電場、磁場、位置、速度を表す。また、$f_s(\vec{r},\vec{v},t)$ は位置－速度位相空間におけるプラズマ粒子の分布関数であり、s はイオンや電子など粒子種を示す。q_s と m_s はそれぞれ電荷と質量を表す。

プラズマ粒子の分布関数は、式(1)で示される通り電磁場によって変形する。電磁場の時空間発展は以下のマックスウェル方程式によって記述される。

$$\nabla \times \vec{B} = \mu_0 \vec{J} + \frac{1}{c^2}\frac{\partial \vec{E}}{\partial t} \tag{2.1}$$

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{2.2}$$

$$\nabla \cdot \vec{E} = \frac{\rho}{\varepsilon_0} \tag{2.3}$$

$$\nabla \cdot \vec{B} = 0 \tag{2.4}$$

ここで $\vec{J}$ は電流密度、ρ は電荷密度、μ_0 は真空中の透磁率、ε_0 は真空中の誘電率、c は光速を示す。ブラソフ方程式(1)を速度空間で積分すると、以下の電荷保存則が得られる。

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \vec{J} = 0 \tag{3}$$

電磁場を変化させるための電流密度 $\vec{J}$ はプラズマの運動によって生じるが、これはブラソフ方程式(1)の第二項にある実空間の流束 $\vec{v}f_s$ を速度空間で積分することによって求まり、電流密度 $\vec{J}$ が電荷保存則(3)を満足する限り、ポアソン方程式(2.3)は自動的に満たされる。以上の方程式は、ブラソフコードにおいて解いているプラズマ粒子の運動論方程式であり、無衝突プラズマの第一原理と呼ぶ。

ブラソフ方程式は最大で実空間 3 次元及び速度空間 3 次元の「超多次元」を扱う方程式であるため、そのままの形で多次元数値積分を行うのは非常に困難であり、またコンピュータで解くには膨大なリソースを必要とする。本研究グループは長年にわたりブラソフシミュレーション手法の開発を行ってきており[2—5]、磁気リコネクション（テアリング不安定性）やケルビンーヘルムホルツ不安定性などのメソスケール現象のみならず[6, 7]、イオンジャイロ半径スケールの半径を持つ非磁化小天体と太陽風との相互作用などのグローバルシミュレーションにも世界で唯一成功している[8—10]。ここでは手法の詳しい解説については省略する。

3. ブラソフコードの並列化

ブラソフシミュレーションでは非常に多くのメモリを必要とするため、並列計算が必須となる。ブラソフコードで使用する物理量は全て格子点上で与えられており、並列化においては領域分割法が有効である。図 1 は実空間 2 次元及び速度空間 3 次元を使用する 5 次元ブラソフコードにおける並列化の概念を示す。我々の目は 4 次元以上の空間を認識できないが、2 次元実空間の各格子上に 3 次元速度空間（速度分布関数）が定義されていると考えると分かりやすい。本研究では図 1 のように実空間（x－y 平面）においてのみ領域分割を行い、速度空間の領域分割は行わない[4]。これは、電荷密度や電流密度などのモーメント量を計算する際に必要な速度空間の積分において、各実空間でのリダクション処理を行わないようにするためである。

5 次元ブラソフコードでは、OpenMP によるスレッド並列も併用している。スレッド並列はそのオーバーヘッドの大きさから、できるだけより外側のループで行うのが効率的である。x86 系 CPU においてはこれまで、スレッド並列を用いないフラット MPI 並列のほうが、スレッド並列と MPI プロセス並列を併用したハイブリッド並列よりも効率的であった。しかし、FX10 や京コンピュータなどの近年の計算機においては、ハイブリッド並列のほうが効率的になる場合がある。また、本研究グループにおける京コンピュータ 6000 ノードの実利用経験より、IO 処理や分散ファイルのデータ解析などの観点からプロセス数をできるだけ減らしたほうが利点は大きい。

ブラソフモデルは 4 次元以上の超次元を扱い、メモリ使用量が非常に多いため、速度空間の格子点を 30^3－60^3 に固定してコアあたりのメモリ使用量 1-4GB に設定しつつ、使用ノード数を増やして計算領域

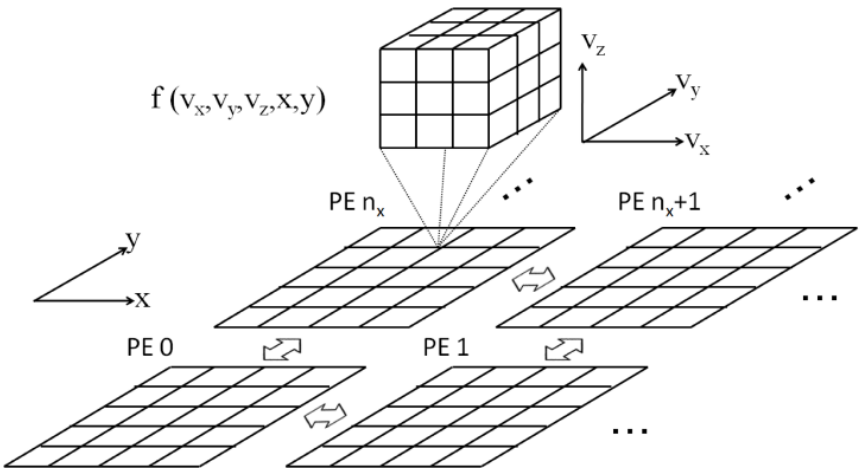

図 1：5 次元ブラソフコードにおける空間領域分割による MPI プロセス並列化[4]。

（実空間の格子数）を拡張していくのが実際の超並列計算機の利用方法である。しかし、近年の計算機においては、ノード内の共有メモリの容量は増えずにコア数のみが増加していく傾向にあるため、単一のループのみをスレッド化する単純な方法には限界がある。

以下に、本研究課題において試した OpenMP スレッド並列化について解説する。5 次元ブラソフコードの計算エンジンは 5 重ループになっており、外側 2 ループ(i 及び j)は実空間(x, y)格子のインデックスを示す。また内側 3 ループ(l, m 及び n)は速度空間(vx, vy, vz)格子のインデックスを示す。OpenMP ディレクティブは最外側ループに挿入する（プログラム 1 参照）。この場合、最外側のループ(j)のみがスレッド並列化される。一方で、!$OMP DO ディレクティブの COLLAPSE オプションは、多重ループのスレッド化を行う機能であり、プログラム 2 の場合は j 及び i のループがスレッド並列化される。

なお、i, j のループと l, m, n のループでは演算やデータの並びが異なるため、COLLAPSE オプションは 2 を指定している。また、流体型の 3 次元コードでは i, j, k のループに対して COLLAPSE オプションを 3 に指定することも可能ではあるが、最内側ループはできるだけループ長を長く取ったほうが効率的であるため、COLLAPSE オプションは 2 のほうがよい。

```
!$OMP PARALLEL DO
DO j = 1,ny
  DO i = 1,nx
    DO n = 1,nvz
      DO m = 1,nvy
        DO l = 1,nvx
           演算
        END DO
      END DO
    END DO
  END DO
END DO
!$OMP END DO
```

プログラム 1：As-Is コード

```
!$OMP PARALLEL DO COLLAPSE(2)
DO j = 1,ny
  DO i = 1,nx
    DO n = 1,nvz
      DO m = 1,nvy
        DO l = 1,nvx
           演算
        END DO
      END DO
    END DO
  END DO
END DO
!$OMP END DO
```

プログラム 2：COLLAPSE を用いたコード

4. 性能測定

本プロジェクトでは、FX10（768 ノード）及び CX400（1476 ノード）を用いたブラソフコードの性能評価を行った。それぞれのシステムにおいて用いたコンパイラオプションは以下のとおりである。

```
FX10: mpifrtpx -Kfast,visimpact,preex,openmp,simd=2 -x250
CX400(富士通コンパイラ): mpifrt -Kfast,openmp,parallel -x250
CX400(インテルコンパイラ): mpiifort -ip -ipo -O3 -xAVX -opemmp
```

ここで、-x250 はユーザ定義関数のインライン化用のオプションである。-Ksimd=2 は SIMD 化強化用のオプションであるが、必ず-Kvisimpact よりも後ろにつける必要がある。なお、FX10 と CX400 ではコーディングの異なるコードを用いて性能測定行った。具体的には、CX400 では As-Is コード（プログラム 3 参照）、FX10 では、整数関数を使用せずに型変換して倍精度実数関数を使用するコードを用いている[11]。プログラム 4 では、整数から倍精度実数への変換が 2 つ、倍精度実数から整数への変換が 1 つ増えているにも関わらず、MAX 関数が倍精度実数関数になっているために FX10 ではこちらの方が高速である。また、性能評価はプログラム 5 に示すように、MPI_Wtime 関数を用いてメインループの計算時間(t3-t2)を計測する。

```
I3 = MAX(I1,I2)
```

プロフラム 3:As-Is コードの例

```
R1 = I1
R2 = I2
I3 = MAX(R1,R2)
```

プロフラム 4: 整数関数を使用せずに型変換して倍精度関数を使用するコードの例

```
Call MPI_Init(ierr)
t1 = MPI_Wtime()
call initialization
t2 = MPI_Wtime()
do time = 1,ntime
      call mainloop
end do
t3 = MPI_Wtime()
call MPI_Finalize(ierr)
```

プロフラム 5: 経過時間の計測の例

単一ノードにおいて、プロセス数とスレッド数を変化させた時の各システム／コンパイラの性能を表 1 に示す。本プロジェクトの計測では、“コアあたり”の格子数を実空間では 40×20、速度空間

では 30×30×30 に設定している。これはコアあたりのメモリ使用量を約 1GB に固定してノード数を増やしていく弱いスケーリングに該当する。CX400 及び FX10 はノードあたり 16 コアであるため、表 1 の測定での実空間の格子数は 160×80 である。なお CX400 ではインテルコンパイラを用いたほうが富士通コンパイラを用いた時よりも約 1.5 倍高速であることから、表 1 の測定ではインテルコンパイラのみで行っており、プログラム 1 と 2（COLLAPSE オプション無・有）の比較を行っている。

まず、FX10 と CX400 において、2 倍近くの性能差がある。これは、クロック周波数及び実効効率においてそれぞれ約 1.4 倍の差があるためである[1,11]。次に、FX10 では 4 スレッド・4 プロセスのハイブリッド並列が最速であり、CX400 では COLLAPSE オプション無の場合はフラット MPI が最速である[1,11]。しかし COLLAPSE オプション有の場合は 4 スレッド・4 プロセスのハイブリッド並列が最速となった。特筆すべき点としては、COLLAPSE オプション有の場合にスレッド数を増やしていっても性能劣化が小さいことが挙げられる。これは、プロセス数を減らすという観点において重要である。一方で、スレッド数 1（フラット MPI）及びスレッド数 2 の場合において、ループの分割の観点からは全く同じにもかかわらず、FX10 と CX400 の両方において、COLLAPSE オプション無・有で性能に差が出た。

次に、CX400 において全ノード計測を行った結果を表 2 に示す。富士通コンパイラとイオンテルコンパイラの両方を使用して、COLLAPSE 無の As-Is コードでフラット MPI(1 スレッド・16 プロセス／ノード)の場合と、COLLAPSE 有のコードでノードあたり 4 スレッド・4 プロセスの場合を比較した。MPI_Init/Finalize の経過時間は、ジョブスクリプトの#PJM -m s オプションで送信されるジョブ統計情報(ELAPSE TIME (USE))から逆算した。計測結果より、インテルコンパイラを用いた場合に主に MPI_Init に大量の時間を費やしており、ジョブの Walltime (10 分を想定)を超過して計測に失敗していたことが判明した。この現象は富士通コンパイラを用いた場合はほとんど発生せず、富士通のジョブスケジューリングシステムとインテルコンパイラ環境の相性の悪さが原因であると考えられる。また、

表 1：ブラソフコードの単一ノード性能と COLLAPSE オプション有・無の比較。

格子点数 160x80x30x30x30/core、時間ステップ 5 でのメインループの計算時間(t3-t2)。

スレッド数／プロセス数	FX10 As-Is ／ COLLAPSE	CX400 As-Is ／ COLLAPSE
1／16	206.64 sec ／ 206.59 sec	107.03 sec ／ 107.82 sec
2／8	199.68 sec ／ 200.16 sec	113.16 sec ／ 107.48 sec
4／4	199.47 sec ／ 199.53 sec	107.75 sec ／ 106.07 sec
8／2	201.08 sec ／ 200.77 sec	113.16 sec ／ 109.02 sec
16／1	201.90 sec ／ 200.83 sec	113.39 sec ／ 108.89 sec

表 2：CX400 におけるブラソフコードの全ノード性能。

格子点数 160x80x30x30x30/core、時間ステップ 5 での計算時間(t2-t3)。

各セクションの 経過時間	1 スレッド per ノード(As-Is) Fujitsu ／ Intel	4 スレッド per ノード(COLLAPSE) Fujitsu ／ Intel
MPI_Init / Finalize	17.04 sec ／ 1153.53 sec	15.04 sec／ 402.45 sec
初期化(t2-t1)	135.33 sec ／ 706.59 sec	13.63 sec ／ 21.42 sec
メインループ(t3-t2)	166.35 sec ／ 124.88 sec	162.33 sec ／ 120.13 sec

As-Is コードを用いてフラット MPI で計測した場合、初期化ルーチン、特に電磁場を平衡解に収束させるために用いている MPI_Allreduce において時間がかかっており、特にインテルコンパイラを用いた場合に顕著であった。重要な結果として、MPI_Init や MPI_Allreduce の経過時間はハイブリッド並列化によりプロセス数を減らすことによって大幅に削減できることが挙げられる。またメインループの自体も、プロセス数を減らすことによって計算効率の向上が若干見られた。

最後に、1 コアあたり 1GB にメモリ使用量を固定したときの、5 次元ブラソフコードの弱いスケーリング性能を図 2 及び図 3 に示す。図 2 はコア数に対する実効速度（GFlops）を表し、図 3 はコア数に対する実効並列効率（スケーラビリティ：N コアの実効効率を 1 コアの実効効率で割った量）を表す。本計測では、COLLAPSE 有のコードを用いてスレッド数 4 のハイブリッド並列で計算した結果を用いている。FX10 においても CX400 においても、全体的な性能は向上しており、実効性能が約 1%上昇している。FX10 においては 98%以上のスケーラビリティが維持されている。また CX400 においても、昨年度の計測では 1024 ノードにおいてスケーラビリティが約 70%に低下していた(イオンテルコンパイラを用いた場合)のに対して、今回は 1476 ノードを用いた場合でも 84%以上を維持している。また富士通コンパイラを用いた場合は、1476 ノードを用いた場合でも 90%以上を維持している(昨年度は 1024 ノードで 85%のスケーラビリティ)。

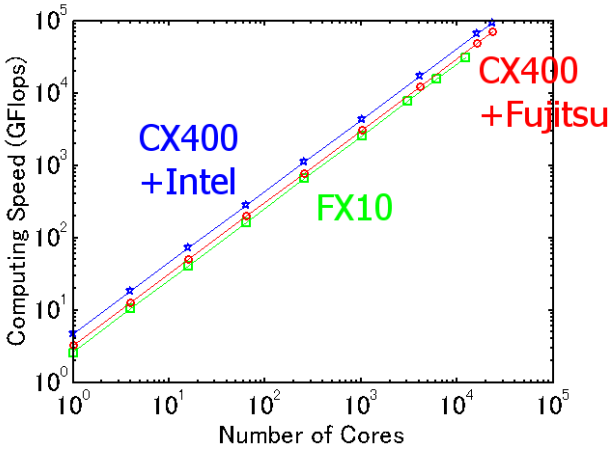

図 2：1GB/core 使用時の 5 次元ブラソフコードの弱いスケーリング性能(1)。コア数に対する実効速度を示している。

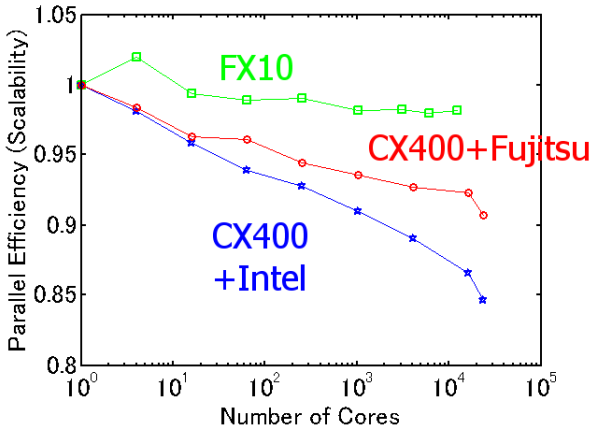

図 3：1GB/core 使用時の 5 次元ブラソフコードの弱いスケーリング性能(2)。コア数に対する実効並列効率を示している。

5. まとめ

本研究プロジェクトで得られた成果は以下のとおりである。

1．OMP DO ディレクティブの COLLAPSE オプションにより、特に多重ループが特徴的な流体型コードなどの場合にはスレッド並列時の性能劣化が削減できる。またこれにより、フラット MPI よりもハイブリッド並列のほうが高速になる可能性がある。本研究のコードでは、ノードあたり 4 スレッドの場合が最速であった。

2．ハイブリッド並列によってプロセス数を減らすことにより、MPI の全体通信に関わる関数の経過時間を削減できる。またこれにより、特にノード数が非常に大きい場合の並列計算において、コードの性能向上が期待できる。

ブラソフコードの全ノード計測において、FX10 の 768 ノード（12,288 コア）を用いた場合の最高性能は 31.0TFlops（実効性能 17.0%）であり、CX400 の 1476 ノード（23,616 コア）を用いた場合の最高性能は 93.6TFlops（実効性能 18.3%）であった。昨年度の計測に比べて並列性能の劣化が抑えられて全体的な性能も向上しており、これは `COLLAPSE` オプションを用いてハイブリッド並列し、プロセス数を減らしたことに依るところが大きい。

また本研究プロジェクトにより、富士通のジョブ管理システムにおいてインテルコンパイラ環境を用いた場合に、富士通コンパイラ環境とスクリプトの記述方法が若干異なる、`MPI_Init` に時間がかかり過ぎる、等の不具合が見受けられ、これらについては改善を期待する。

参　考　文　献

[1] 梅田 隆行, 深沢 圭一郎: 第一原理無衝突プラズマシミュレーション用 5 次元ブラソフコードの性能評価, 平成 24 年度先端的計算科学研究プロジェクト報告書, 2013.
http://www2.cc.kyushu-u.ac.jp/scp/users/forum/forum20130426/04_umeda.pdf

[2] Umeda, T.: A conservative and non-oscillatory scheme for Vlasov code simulations, *Earth Planets Space*, Vol.60, No.7, 773—779 (2008).

[3] Umeda, T., Togano, K., Ogino, T.: Two-dimensional full-electromagnetic Vlasov code with conservative scheme and its application to magnetic reconnection, *Comput. Phys. Commun.*, Vol.180, No.3, 365—374 (2009).

[4] Umeda, T., Fukazawa, K., Nariyuki, Y., Ogino, T.: A scalable full electromagnetic Vlasov solver for cross-scale coupling in space plasma, *IEEE Trans. Plasma Sci.*, Vol.40, No.5, 1421—1428 (2012).

[5] Umeda, T., Nariyuki, Y., Kariya, D.: A non-oscillatory and conservative semi-Lagrangian scheme with fourth-degree polynomial interpolation for solving the Vlasov equation, *Comput. Phys. Commun.*, Vol.183, No.5, 1094—1100 (2012).

[6] Umeda, T., Togano, K., Ogino, T.: Structures of diffusion regions in collisionless magnetic reconnection, *Phys. Plasmas*, Vol.17, No.5, 052103(6pp.) (2010).

[7] Umeda, T., Miwa, J., Matsumoto, Y., Nakamura, T. K. M., Togano, K., Fukazawa, K., Shinohara, I.: Full electromagnetic Vlasov code simulation of the Kelvin-Helmholtz instability, *Phys. Plasmas*, Vol.17, No.5, 052311(10pp.) (2010).

[8] Umeda, T., Kimura, T., Togano, K., Fukazawa, K., Matsumoto, Y., Miyoshi, T., Terada, N., Nakamura, T. K. M., Ogino, T.: Vlasov simulation of the interaction between the solar wind and a dielectric body, *Phys. Plasmas*, Vol.18, No.1, 012908(7pp.) (2011).

[9] Umeda, T.: Effect of ion cyclotron motion on the structure of wakes: A Vlasov simulation, *Earth Planets Space*, Vol.64, No.2, 231—236 (2012).

[10] Umeda. T., Ito, Y.: Entry of solar-wind ions into the wake of a small body with a magnetic anomaly: A global Vlasov simulation, *Planet. Space Sci.*, in press (2014).

[11] 梅田 隆行: 京コンピュータを用いた宇宙プラズマの第一原理ブラソフシミュレーション, サイエンティフック・システムズ研究会 科学技術計算分科会 2013 年度会合 (2013).
http://www.ssken.gr.jp/MAINSITE/event/2013/20131023-sci-2/lecture-01/SSKEN_sci2013-2_umeda_paper.pdf