J. Plasma Fusion Res. Vol.89, No.1 (2013) 45-48

講座

核融合プラズマシミュレーションの技法
ー大規模並列計算環境の活用ー

Methods of Fusion Plasma Simulation
− Utilizing Messively-Parallel Computation −

1．はじめに

渡邉智彦
核融合科学研究所・総合研究大学院大学

（原稿受付：2012年11月12日）

本講座では，これから大規模なシミュレーションを始めようとする大学院生や研究者に向けて，核融合プラズマのシミュレーションで用いられる計算技法に関する解説を試みる．この章では，次章以降の本論への序論として，本講座の背景と狙い，並列計算の必要性，並列計算機の変遷，そして並列計算の基礎概念，について述べる．

1.1　本講座の背景と狙い

高性能コンピュータを用いたシミュレーション解析は，今や，理学，工学の多種多様な分野で行われ，日々の天気予報から創薬にいたるまで日常生活にも密接な関連をもつようになった．核融合プラズマ分野においても，その重要性は他の科学・工学分野に勝るとも劣らず，複雑なプラズマ現象の解明や理解，そして将来の実験装置の性能予測に向けて，様々なシミュレーション研究が進められている．本講座では，これから本格的なシミュレーション研究をはじめようとする大学院生や研究者を対象に，核融合プラズマ研究で用いられる計算技法についての解説を試みる．特に，並列プログラミング手法に焦点をあて，その基本的な考え方から実際のコーディング例までをできるだけわかりやすくまとめたい．

一口にコンピュータといっても，スマートフォンから，最先端のスーパーコンピュータまで，その規模や性能は幅広い．2012年末時点で，top500 リストの 3 位にランクされる計算性能をもつスーパーコンピュータ「京」は，その名のとおり 1 秒間に 1 京回もの浮動小数点演算を実行することができる．一方，その構成は非常に大規模かつ複雑なものとなり，効率的なシミュレーション実行のために多くの努力が傾けられている．また，IT 技術の進歩とその日常製品（コモディティ）化に伴う計算機環境の変化のスピードも著しく速い．例えば，現代のスマートフォンのプロセッサであれば 10-20 GFlops 程度の理論演算性能を有しているが，これは，初代地球シミュレータのベースとなった SX-5（1998年）の 1 プロセッサの演算性能（8 GFlops）を超えるほどである．

ハードウェアに比べ，その上で動作するシステムソフトウェアの進展は比較的ゆるやかである．例えば，多くの研究者がシミュレーションに使うプログラミング言語 Fortran90 は1990年に規格化されてから20年以上が経つが，今後もしばらく使われ続けるであろうし，その元となる Fortran77 で記述されたコードもしばしば用いられている．また，並列処理計算の基盤となる MPI（Message Passing Interface）ライブラリ[1,2]も，1995年に ver 1.2 が策定された後，いくつかの拡張はなされているものの核心部分は保たれたまま多くのアプリケーションで利用されている．広く普及したシミュレーションや理論モデルは，さらに長い寿命をもつ．例えば，1960年代初頭にJ.Dawson等によって始められたプラズマの粒子シミュレーション（PIC: Particle-in-Cell）法は，その原型をとどめたまま現在も多くの研究で用いられている．

このように，ハードウェア，システムソフトウェア，アプリケーションは，それぞれ異なった時間スケールで進歩するため，ある程度先進的なシミュレーション研究においては，計算モデルをどのように計算機環境に最適化するかという問題に向き合うことになる．特に1990年代半ば以降，単一の演算プロセッサの性能向上に限界が見えてきてから，並列計算機が登場し，これに伴い計算機ハードウェアの変化はより加速した．また，計算機環境が大きく変化

1. Introduction

WATANABE Tomo-Hiko
author's e-mail: watanabe.tomohiko@nifs.ac.jp

し続けると，ひいては，シミュレーション・モデルの変更を促すことにもなろう．こうした状況の中，上述の3つの技術（ハードウェア，システムソフトウェア，アプリケーション）をうまく調和させることが，先端的な大規模シミュレーションの実行には不可欠なものとなった．

大規模シミュレーションで用いる技法は，「ある時点で利用可能な計算機環境下において，具体的な研究課題に関わる問題を最適化する」方法であるとも言え，一般的な数値計算の教科書にはいくつかの例がありつつも，普遍的な体系化がなかなか追いつかないのが現状であろう．一方，大規模並列計算に向いたアルゴリズムの選択や並列実行効率の向上，通信の最適化のための工夫などの様々な「技法」について，シミュレーションの研究論文を読んだだけでは，その細部を知るのは困難である．こうした事情から，大規模並列コンピュータを実際のシミュレーション研究にどのように活用すればよいか，入門者にはやや敷居が高く感じられるかも知れない．そこで本講座では，実際に核融合プラズマ研究に用いられているコードを題材に，非並列・チューニング前の書き方からはじめ，どのように大規模並列環境に適した形にコードを書き換えていくか，その過程と背景にある考え方と合わせ，段階を追った説明を試みる．

本講座の構成と執筆者は以下の予定である．

第1章　はじめに（渡邉智彦）
第2章　並列コードを開発するための基礎技法（坂上仁志）
第3章　MHDシミュレーションのコーディング技法（三浦英昭，藤堂泰，後藤俊幸）
第4章　Vlasovシミュレーションのコーディング技法（渡邉智彦，井戸村泰宏）
第5章　粒子シミュレーションのコーディング技法（佐竹真介，内藤裕志）

本論への導入として，並列計算の必要性，並列計算機の変遷，並列計算の基礎概念について，本章の後半で概説する．次章以降では，現在（または近い過去と未来も含むであろうが）の計算機環境の下で核融合プラズマ研究における課題を解決するためのレシピを記述しようとすることになるが，主題の性質からそれらは現時点でのスナップショットとして捉えていただきたい．今後の計算機技術や計算モデルさらにプラズマ理論の発展に伴って，核融合プラズマのシミュレーション技法もともに進化を続けることであろう．

1.2　なぜ並列計算が必要か

これを議論するためには，最近の核融合プラズマのシミュレーション研究課題を対象に，実際にどの程度の計算速度が必要となるかを簡単に見積もってみるのがよいだろう．ここでは，4章で紹介するジャイロ運動論的Vlasovシミュレーションを例にとってみよう．ジャイロ運動論的方程式は，5次元位相空間上で定義される．各座標方向におおまかに100点の空間格子を設定したとすれば，計10^{10}（＝100億）点となるが，各方向に少しずつ節約して，10^9（＝10億＝1 G）点の格子上で分布関数の時間発展を計算することにしよう．少なく見積もっても，時間積分の1段あたりおよそ100回の浮動小数点演算を行い，さらに4段（4次）の時間発展スキームを使うと，時間ステップあたり4×10^{11}回の演算が必要となる．典型的なドリフト波の成長率γは，イオンの移動時間（transit time）τで規格化しておよそ0.1程度であり，不安定性の成長から飽和さらに乱流遷移に至るまでに成長時間の25倍は計算するとすれば250τの時間計算を行う必要がある．一方，十分な空間解像度を得るには波長の短い揺動を扱う必要があり，そのため時間ステップ幅は0.01τ程度に設定するとしよう．すると全計算ステップ数は25,000となり，全体の演算量は10^{16}となる．これを1秒間に1兆回（10^{12}）の浮動小数点演算（1 TFlops）が実行可能な計算機があれば，およそ10,000秒でシミュレーションを実行することができる．実際のシミュレーションでは，より高解像度が求められ（例えば3次元ヘリカル磁場構造をもつ核融合プラズマ閉じ込め装置であるLHDの解析では500億以上の格子点を使い，かつ，より細かな時間ステップ幅が要請される[3]），また，多成分プラズマやグローバルな形状効果が加わるなど，さらに大きな計算資源を必要とする場合が多い．

さて，上で述べた大規模シミュレーションで用いる計算機はどのような形で実現され得るだろうか？文献[4]にあげた教科書に興味深い見積もりがあるので，エッセンスをそのままに，上記のケースへと例えを変えて紹介する．上述のLHDプラズマの解析に使った500億個の要素をもつ配列x(i)，y(i)，z(i)に対して，従来型のフォンノイマン型計算機において以下のdoループを実行することを考える．

```
do i = 1, 50000000000
  z(i) = x(i) + y(i)
end do
```

この処理を逐次的に実行するには，1.5×10^{11}個のデータをメモリとの間で移動させなくてはならない．一方，データ転送速度は光速度（3×10^8 m/sec）で制限されているので，上記の処理を1秒間で行うには（すなわち，実行性能で5×10^{10} Flops=50 GFlopsを達成するには，1.5×10^{11}回の転送が発生するので），全メモリへの平均距離を2×10^{-3} m以内に近づける必要がある．そのためには，1.5×10^{11}個のデータをおよそ4×10^{-3} m四方の中に収めることになり，データを表す1ワード間の平均距離を，$4\times10^{-3}/(1.5\times10^{11})^{0.5}$ m＝10^{-8} m＝10 nm以下にする必要がある．すなわち，Siの共有結合半径（およそ0.1 nm）の100倍程度の距離の中に1ワード（倍精度で64ビット）のメモリ素子を実装しなければ，仮にどんなに高速な演算プロセッサが実現できたとしても上記の処理を1秒間で完了することはできないことになる．したがって，この問題に対する解決策は，データ転送と計算処理を並列で行うことにあることは自明であろう．

1.3　並列計算機の変遷

では，並列処理を行うコンピュータとはどのようなものだろうか．そのアーキテクチャの詳細については次章に譲

り，まず，ここでは我が国での利用を中心に並列計算機の変遷について概説しよう．

1980年代には，日本製のベクトル型計算機が世界を席巻し，米国との貿易摩擦を引き起こすほどであった．従来型のベクトルマシンは，現在はNECのSX-9のみであるが，当時はCray，富士通，日立の各社が激しい開発競争を進めていた．ベクトル型計算機の特徴は，パイプラインによってデータ処理を一斉に行えることにあり，優れたコンパイラと相まって，ある程度のコツさえおぼえれば，利用者はそれほど多くの負担を感じずにスーパーコンピュータ利用の恩恵にあずかることができた．

しかし，計算機の性能向上に対する要求は高まりこそすれ衰えることはない．1990年代に入ると単一の演算プロセッサのみでは，その要求を満たせなくなり，1990年代中頃には各社が相次いで並列計算機を開発しはじめた．これに少し先立って，OSにUnixが採用され，ユーザ環境の標準化が徐々に進行した．日本では，独自プロセッサを積んだ富士通のVPP500，NEC SX-4，日立SR2201などがある．一方，米国では，本来はワークステーション用であったRISCプロセッサを多数組み合わせたシステムが開発された．原研にいち早く導入されたParagonなどもその流れにある．

当時は，並列計算機利用のためには各社独自のライブラリを利用してノード間通信や並列処理の制御を行っていた．しかしこれではアプリケーションの汎用性が失われ，開発効率が低下する．そのためシステムソフトウェアの規格が作られ，並列処理の制御と計算ノード間の通信を行うライブラリとしてMPIが標準的に用いられるようになった．

国内において，数千にもおよぶプロセッサを使う超並列計算機が広く利用されるようになったのは，2002年の初代地球シミュレータ（ES）の登場以降であろう．ESは，1ノードに8つのベクトルプロセッサを持ち，ノード内では共有メモリを使ったスレッド並列処理ができた．さらに640の計算ノード（それぞれが一つの筐体に収められていた）を単段のクロスバーネットワークで接続した強力なマシンであり，2年半にわたってTop500リストで一位の座を保った．米国をはじめとした海外では，この間，プロセッサのコモディティ化がさらに進行した．インテルのx86プロセッサやAMDのOpteronを用いた大規模マシンが相次いで開発された（東京大学・京都大学・筑波大学で導入されたオープン・スパコンT2Kは，この流れの延長上にあるといえよう）．

しかし，従来型のCPUプロセッサを多数搭載しただけでは，電力や設置スペースの制限から自ずと限界がある．そのため高密度実装の流れが2000年代中頃から始まる．その先駆けは，IBMのBlue Geneであろう．個々のプロセッサはそれほど強力でないが，電力消費を抑え高密度実装を実現して，ESを超えるLinpack性能を達成した．さらに2007年には，20万を超える演算コアを搭載したBlue Gene/Lは，470 TFlopsという数値をたたき出した．

高密度実装は，部品点数を減らし，耐故障性を向上させるというメリットも併せ持つ．こうしたトレンドは，市販のプロセッサにもおよび，2008年頃から一つのソケットに複数の演算コアを載せたマルチコア・プロセッサがリリースされ，さらにはメニーコアの時代に入ろうとしている．2011年に日本がTop500リスト首位の座を奪い返した「京」コンピュータは，8つの演算コアを一つのプロセッサに搭載し，それらを88,128ノード接続した巨大なマシンであり，2012年9月末から供用が開始されている．一方，スーパーコンピュータの性能競争は果てしなく，2012年末時時点ではTitan（Cray XK7）が17 PFlopsという性能を挙げてトップを走っている．またここでは触れなかったが，汎用GPUを利用した計算機も開発され，大規模シミュレーションに利用されている．日本では，東京工業大学のTsubame 2.0がよく知られている．

一方，我が国の核融合研究分野で最近利用されている主なスーパーコンピュータとしては，大阪大学のSX-9，日本原子力研究開発機構のBX900，核融合科学研究所のSR 16000/M1，国際核融合エネルギー研究センターのHelios（Bullx B510）などがある．いずれも大規模なシステムであり，特に後の2者は100 TFlops級の大規模計算を実行可能であることから，核融合プラズマシミュレーションを大きく進展させる原動力になると期待される．

1.4 並列計算の基礎概念

以下では，並列計算がどのように行われるか，その基礎概念を解説し，講座の導入部としての本章を終えたい．

現在，一般的に行われている並列計算は，大きく二つの種類に分けられる．

一つは，メモリ空間を共有する複数の演算プロセッサ（コア）が計算処理を分担し，データの読み書きを同一のメモリ領域に対して行う，共有並列である．プログラムの並列化は，コンパイラの自動スレッド並列機能を用いて行うか，OpenMPを利用した指示行によって行われる．そのため，多くの場合，プログラムの構造を大きく改変する必要はなく，比較的簡便な方法である．ただし，この場合の並列計算による性能向上は，共有並列可能なプロセッサ・コア数の上限で決まり，4から16ないし32並列程度が現状の上限である．

もう一つのタイプは，それぞれの計算ノードで異なるデータを保持し，その処理を並列で行う分散並列である．異なるノード間でのデータのやりとりや，並列計算のタイミングを制御するために，MPIライブラリが用いられる．このため，シミュレーション・コードは一般に多くの変更を必要とし，並列計算の入門者にはやや敷居が高く感じられるであろう（MPIを使った並列プログラミングの困難さを緩和するために，HPFなどいくつかの言語環境が開発されており，それらについては次章で触れられる）．一方，並列数の上限は，システムを構成する全ノード数（または全演算コア数）まで可能であり，うまくプログラムを作ることができれば，大きな並列化効果が期待できる．

分散並列プログラムは，また，2種類のカテゴリーに分けられる．一つは，マスター・スレーブ型とよばれ，主プ

ロセスが複数の子プロセスの実行を管理するものである．これは，各並列プロセスの独立性が高く，プロセス毎の実行時間がばらつき，同期が不要な場合に特に有用であり，プロセス毎にまったく異なる計算を実行させることもできる．例えば，異なる温度をもつ熱浴の下で，分子動力学計算を実行する場合などが挙げられる．もう一方は，単一のプログラムを，複数の並列プロセスで同期を取りながら同時に実行する方法である．並列プロセス番号（ランク）に応じて，分割したデータのどの領域を計算するかを指定し，他の領域とデータをやりとりしながらシミュレーションを進める．領域分割された流体シミュレーションがその代表的な例である．

前節でみたように，現代の並列計算機の利用には，共有並列と分散並列を両方とりいれたプログラミング技法が有効とされる．さらにプログラムの実行性能を高めるには，キャッシュの有効利用や演算パイプラインの利用が必要であり，さらにはネットワーク形状を考慮したプロセス配置や計算と通信のオーバーラップなどが行われることもある．また，次世代のエクサスケール・コンピュータ開発に際しては，計算科学とアプリケーション開発者が協業したco-designが議論されており，大規模並列計算のためのプログラミングは，さらに困難かつ複雑なものになっていくかもしれない．こうしたなかで，本講座が，これから核融合プラズマシミュレーションの最先端に挑戦しようという学生，若手研究者諸賢のために，何らかの役に立つことを願いつつ本章を終えたい．

参考文献

［1］http://www.mpi-forum.org/
［2］W. Gropp *et al., Using MPI* (MIT Press, Cambridge, 1994).
［3］T.-H. Watanabe *et al.,* Phys. Rev. Lett. **100**, 195002 (2008).
［4］P. パチェコ：MPI 並列プログラミング（培風館，2001）．

渡邉智彦（わたなべ ともひこ）

気がつけば，シミュレーションをやるようになってからの人生の方が，その前よりも長くなりました．次はどんなスーパーコンピュータを使えるようになるのか楽しみに，そろそろ，また新しい研究を始めたいと思っています．