プラズマ粒子シミュレーションコードの並列化モデルと速度向上率の評価

上岡 功治･村田 健史　愛媛大学工学部情報工学科
上田 裕子　宇宙開発事業団
岡田 雅樹　国立極地研究所
大村 善治･松本 紘　京都大学宙空電波研究センター

本研究では，プラズマ粒子コードの並列化手法として，粒子分割法と空間分割法を組み合わせたハイブリッド分割法を提案する．ハイブリッド分割法では，一つの基礎方程式を、一つのサブルーチンで構成する．粒子計算サブルーチンを粒子分割法により並列化し，流体計算および電磁場計算サブルーチンを空間分割法により並列化する．各サブルーチンの並列化数を個別に設定することにより，最高の速度向上率を得る手法である．本研究では，MPI 通信関数を用いて，プラズマ粒子コードの一つであるプラズマ粒子流体混成コードに適用した。その結果，128x128 格子点，格子点あたりの粒子数 320 の場合，8 並列で 7.089，16 並列で 10.885 の速度向上率を得た．

A Parallel Computing Method for Plasma Particle Simulaton Code

Koji Ueoka and Ken T. Murata: Faculty of Engineering, Ehime University
Hiroko O. Ueda: NASDA System Analysis and Software Laboratory
Masaki Okada: Information Science Center, National Institute of Polar Region
Yoshiharu Omura and Hiroshi Matsumoto: RASC Kyoto University

In the present paper, we propose a high performance parallelization method for plasma particle codes. Particle simulation codes are composed of two parts: particle calculation parts and electromagnetic field parts. The technique is based on combination between data parallelization for particles and area parallelization for fields. We applied this technique to a plasma hybrid code. The parallel number is independently set in each subroutine in order that the code obtains a best computing performance. The best speed-up ratio was 7.089 in 8 parallelization case, and 10.885 in 16 parallelization case, respectively. Therein, the numbers of parallelization in particle solving subroutine and fluid solving subroutine were maximum number of PE (8 or 16), and electromagnetic solving subroutine was not parallelized, respectively.

1.　はじめに

現在，高速演算や大容量メモリを必要とする様々な分野で，並列計算機の実用的利用が始まっている．物理化学のコンピュータシミュレーションのうち，様々な媒質の振る舞いを調べる方法として，媒質の運動をつかさどる微積分方程式を数値的に解く流体シミュレーションや粒子シミュレーションが有効である．流体シミュレーションは，媒質を流体として捉え，一方，粒子シミュレーションは，個々の粒子に働く力を求め，粒子の運動を数値的に解く．

本研究では，粒子分割法と空間分割法を組み合わせたハイブリッド分割法を提案する．さらに，プラズマ粒子コードにこの手法を適用し，その計算効率について調べる．

2.　ハイブリッド分割法の提案

2.1.　プラズマ粒子シミュレーションコードの概要

一般に，プラズマ粒子コードは，粒子計算と流体

計算, および電磁場の計算より成る. 粒子計算および流体計算は, 各媒質の運動(量)方程式や連続の式, エネルギーの式などを解く. 一方, 電磁場の計算は Maxwell 方程式を解く.

プラズマの粒子シミュレーションでは、空間を離散的な格子に分割する. 特定の格子内にある粒子は, 近傍の格子点上で定義された電磁気力のみを受け, 粒子間力は考えない.プラズマ粒子シミュレーションでは, このようにして得られた個々の粒子の電荷や速度を積分して, 電荷分布や電流分布が求められる. 粒子より求められた電荷や電流により, Maxwell 方程式を解き, 電磁場を時間方向に解き進める.

2.2. ハイブリッド分割法の概要

プラズマ粒子コードは, 複数のサブルーチンの組み合わせにより構成される. 一つの基礎方程式は, 1 つのサブルーチンに対応する. ハイブリッド分割法では, サブルーチンごとに異なる分割法を採用する. すなわち, 粒子計算のサブルーチンでは粒子分割を行い, 電磁場や流体計算のサブルーチンでは空間分割を行う.

プラズマ電磁粒子コードでは, 同じ基礎方程式系で, 異なるコードが存在する. これは, 方程式の差分化法やアルゴリズム, また, 各サブルーチンルーチンの計算手順などが, コードによって異なるからである. また, 並列化のためにアルゴリズムそのものを変更するのは, 望ましくない. したがって, ハイブリッド分割法の実装は各コードによって異なり, 一般化は容易ではない. 一般的には, 以下のような手続きで実装する.

(1) タイムチャートを作成し, 各サブルーチンの計算順序を決める.
(2) 各サブルーチンを, 粒子分割または空間分割により実装する.
(3) 各サブルーチン間の関係を調べ, それに応じたデータ通信を実装する.
(4) 各サブルーチンの速度向上率を独自に調べる.
(5) コード全体の速度向上率が最大となるよう各サブルーチンの並列化数を決定する.

ハイブリッド分割法は, 粒子計算部と電磁場および流体計算部とで処理が異なる. サブルーチンごとに, 最も計算時間が短くなるような並列化数を, あらかじめ求める. これにより, サブルーチンごとに, 異なる並列化数により並列計算を行う.

3. ハイブリッド分割法のプラズマ電磁粒子コードへの適用

本節では, 前節で提案したハイブリッド分割法を, 既存のプラズマ粒子シミュレーションコードに適用する. 本稿では, 例として, プラズマ粒子シミュレーションコードの一つである, プラズマ粒子流体混成コードを用いる. また, PE 間の通信ライブラリとして, MPI(Message-Passing Interface)を用いる. MPI は, 現在, 多くの並列計算機で標準化されている代表的な通信ライブラリである. なお, 提案するハイブリッド分割法は, 一般的なプラズマ粒子コードであれば, 適用可能である. また, MPI 以外の通信ライブラリでも実現が可能である.

本節では, 各サブルーチンの並列化数を決定する方法について述べる. 一般に, 並列計算では, 逐次計算量が少ないほど計算効率が高い。しかし, 後に示すように, ハイブリッド分割法では, すべてのサブルーチンを, 最大の PE 数で並列化しないほうが, 速度向上率がよい場合がある. したがって, サブルーチンごとの最高速度向上率を与える並列化数を評価するための定式が必要となる. 例として, プラズマ粒子流体混成コードにおける定式化を行う.

本稿では, 各々のサブルーチンに対して並列化数 n_i を, そのサブルーチンの全計算時間を $T_i(n_i)$ とする. ここで, i は, サブルーチンを識別する引数である. 電場計算, 磁場計算, 粒子計算, 電子流体計算, および電流密度計算の各サブルーチンを, それぞれ E,B,B,P,e,および J とあらわす. したがって, 全計算時間 T は, 各サブルーチンの計算時間の総和として, 次式で表される.

$$T = \sum_{i}^{E,B,P,e,J} T_i(n_i)$$

次に, サブルーチン i の全計算時間 $T_i(n_i)$ について考える. サブルーチン i の全計算時間は, 実計算時間 $T_i^{cal}(n_i)$, および PE 間でのデータ通信時間 $T_i^{com}(n_i)$ で与えられる.

$$T_i(n_i) = T_i^{cal}(n_i) + T_i^{com}(n_i)$$

一般に, プラズマ粒子計算では, PE 間の同期を取る時間は, 実計算時間やデータ通信時間と比較して小さい. したがって, 本論では, PE 間での同期時間は無視する. 各サブルーチンの実計算時間を定式化する. プラズマ粒子シミュレーションでは, 各サブルーチン内での計算のほとんどが, 100%の効率で並列化可能なループ計算になっており, 逐次計算部分はごくわずかである. すなわち, n_i 並列のサブ

ルーチン i の実計算時間 $T_i^{com}(n_i)$は,

$$T_i^{cal}(n_i) = \frac{T_i^{cal}(1)}{n_i}$$

となる.

各サブルーチンのデータ通信時間について考察する. 通信時間(通信速度)は, 通信を行う PE 数, 通信量に依存する. これらは, 並列計算機の内部通信ロジックなどにより異なるため, 定式化が容易ではない. 本稿では, ベンチマークテストにより, あらかじめ通信関数ごとの通信時間を測定し, これを用いて速度向上率の評価を行う方法を提案する. 各通信関数のベンチマークをあらかじめ測定することにより, サブルーチン i の通信時間 $T_i^{com}(n_i)$を評価することができる.

速度向上率は, T をもっとも小さくする $T_i^{com}(n_i)$の組で表される. 一般に, これは, 各サブルーチンの計算時間 $T_i^{com}(n_i)$を最小にする n_i の組で与えられる. しかし, プラズマ粒子計算では, サブルーチンごとの並列化数を独立に決定できないことがある. このような依存関係は, コードによって異なるため, 一般化することは容易ではない. したがって, コードのアルゴリズムを考慮して, 最小の計算時間を求める n_i を求めることになる.

4. プラズマ粒子流体混成コードの速度向上率

4.1. 速度向上率の測定環境

本節では, プラズマ粒子流体混成コードによる実験を行う. 実験は, 与えられた格子点数や粒子数などのシミュレーションパラメータに対して, 最高の速度向上率を得ることを目標とする. 以下では, 格子点数が 128x128 格子点数の 2 次元で, 粒子数が(a)2,621,440 個, (b)5,242,880 個, (c)10,584,760 個(1 格子あたりの粒子数 160, 320, 640 個)をパラメータとして与えた場合の, 速度向上率の評価を行う.

本研究で実測値評価のために用いた並列計算機は, 富士通社製の並列計算機AP3000(以下AP3000)である. 通信関数として, MPI 関数を用いた. コンパイラは mpifrt で, オプティマイズ等は行っていない.

AP3000 は, 一つのノードが 2 つのプロセッサから構成される共有分散メモリ型の並列計算機である. 本研究で用いた AP3000 は, それぞれのノードに 640Mbyte のメモリが備わっている. 本研究では, 実行時のオプション設定により, 16PE の分散メモリ型並列計算機として設定し, 測定を行った.

4.2. 並列化数決定のための予備測定

まず, 上記のパラメータにおける, 各サブルーチンの実計算時間の実測値を求めた. さらに, ベンチマークプログラムを用いて, 用いる MPI 関数の通信速度特性を求めた. ベンチマークプログラムは, Pallas 社の b_eff Benchmark (version 3.3)を用いた. プラズマ粒子流体コードで用いる MPI 関数は, (a) MPI_SENDRECV 、 (b) MPI_ALLREDUCE 、 (c) MPI_BCAST の 3 つである. ベンチマーク測定結果を, 図1に示す. それぞれのデータ通信量は, (a) 2kbyte、(b) 128kbyte、(c) 8kbyte から 64kbyte となる. すなわち, MPI_ALLREDUCE では、ほぼ band width の速度が達成されており、速度向上の余地はない. MPI_SENDRECV と MPI_BCAST では、より規模の大きな計算では, 速度向上の可能性がある.

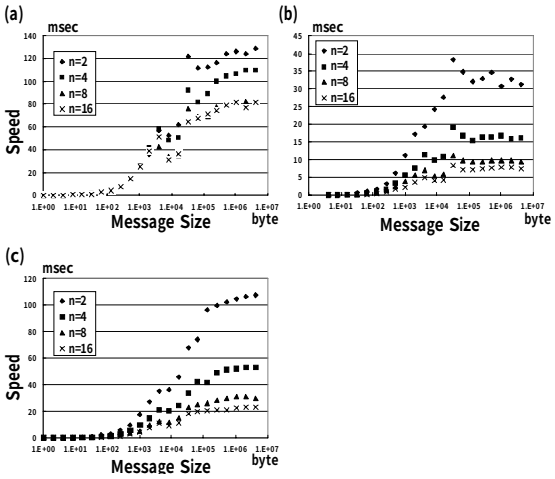

図 1 b_eff による AP3000 ベンチマークテストの結果 : (a)MPI_SENDRECV , (b)MPI_ALLREDUCE , (c)MPI_BCAST

実計算時間とデータ通信時間を, それぞれのサブルーチンごとにまとめた結果を, 図2に示す. これらの図より, 各並列化数における通信時間を評価することができる. 図は, 2, 4, 8, 16 並列での全計算時間を示したものである.

図2(a)と図2(b)は, それぞれ, 電場, 磁場計算サブルーチンの計算時間である. 電磁場量は, 粒子計算のために, 各 PE が全ての空間領域の値を持つ必要がある. そのため, MPI_BCAST によるデータ通信が必要となる. 電磁場計算時間は, 通信時間に比較して短いため, これらのサブルーチンでは, 逐次計算によって最小の計算時間が得られる.

図2(c)は, 流体計算サブルーチンの計算時間で

ある. 一般的な空間分割法と同様に, のりしろ領域のデータ交換のみ必要となる. そのため, 通信時間が他の通信と比較して短い.

図2(d)は, 粒子計算サブルーチンの計算時間である. 一般に, 粒子計算では, 通信時間と比較して, 実計算時間が大きい. そのため, 最大並列化数で全計算時間が最小となる事が多い. 粒子計算では, 実計算時間は粒子数によって決まり, 通信データ量は格子点数によって決まる. したがって, 同じ格子点数でも, 粒子数が比較的小さい場合には, 最大並列化数が最小計算時間とはならない.

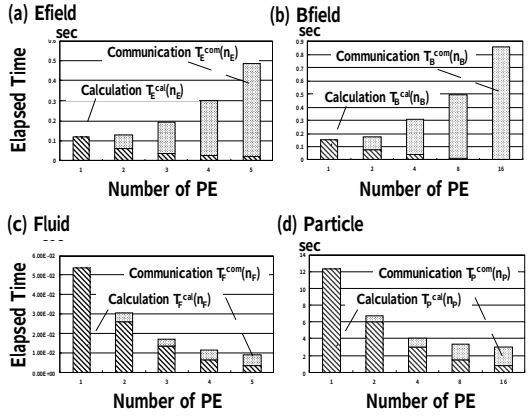

図2　AP3000 によるハイブリッドコードの各サブルーチンの計算時間とデータ通信時間：(a)磁場計算サブルーチン，(b)電場計算サブルーチン，(c)流体計算サブルーチン（電子流体計算と電流密度計算），(d)粒子計算サブルーチン

4.3 速度向上率の測定

4.3.1　全てのサブルーチンで並列化数が等しい場合

まず, 全てのサブルーチンが等しい並列化数 n_{MAX}(=16)で並列化される場合について考える. この時, 各サブルーチンの計算時間の式に n_i=16 を代入する事により, 計算時間および通信時間を算出することができる.

AP3000 を用いた測定結果を, 図 3 に示す. 図の(1), (2)、(3)は、それぞれ粒子数 2,621,440、粒子数 5,242,880、粒子数10,584,760 の速度向上率である. 格子点数は, すべて 128x128 である。図より, 粒子数が増えるほど, 速度向上率が上昇することがわかる. 8 並列, および 16 並列における速度向上率の実測値は, 6.748 および 9.552 となった.

4.3.2　サブルーチン毎に並列化数を設定する場合

本節では, サブルーチンごとに並列化数を設定することにより, 4.3.1 で得られた速度向上率の向上を試みる. そのために, 粒子計算, 電場計算, 磁場計算, 流体計算の並列化数を, 独立に決定する. 図 5 によると, 各ルーチごとの, 最高速度向上率を与える並列化数は, 12, 1, 1, 12である. したがって, これらの並列化数により, 全計算の最高速度向上率を得ることができる.

粒子数10584760、格子点数128x128 のパラメータで, AP3000 を用いて実測を行った結果を、図 8(4)に示す。8 並列, および16 並列における速度向上率の実測値は, 7.089 および 10.885 となったこれらは, 4.3.1 で得られた値よりも大きいため, サブルーチン毎に並列化数を変更する手法が有効であることを示している. すなわち, サブルーチン毎に並列化数 n_i を設定する事により, 単純に PE 数で並列化を行うよりも高い速度向上率を望むことができる.

4.3.3　速度向上率の考察

図 3 によると, 8 並列, 16 並列の最高速度向上率は, 7.089および10.885であった. 8並列の速度向上率は比較的高いが, 16 並列の速度向上率は, 十分とはいえない. これは, 全計算時間のうち占める割合が最も高い粒子計算において, 実計算時間とデータ通信時間が逆転することによる(図 2(d)). すなわち, より高い速度向上率を得るためには, 1 格子あたりの粒子数が多いほど, ハイブリッド分割法による計算効率は高くなると言える.

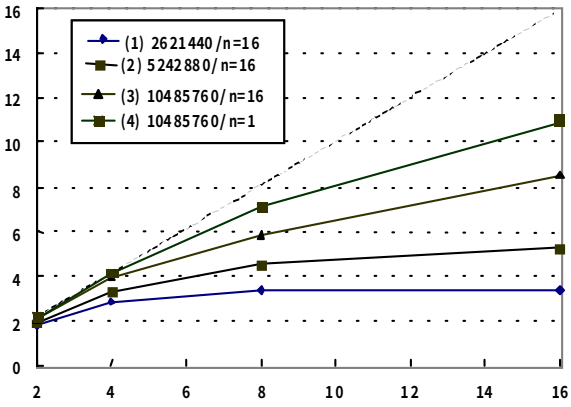

図 3　プラズマ粒子流体混成コードにおける速度向上率の比較：(1)粒子 2,621,440 個/場を並列化した場合，(2)粒子 5,242,880 個/場を並列化した場合，(3)粒子 10,584,760 個/場を並列化した場合，(4)粒子 10,584,760 個/場を逐次計算した場合