A 2 1 4 CUDA によるプラズマ 2 流体不安定性計算の加速

小川 慧[○](東工大院), 青木 尊之(東工大)Acceleration by GPU computing for Simulation of
Two-Stream Plasma Instability
Satoi OGAWA and Takayuki AOKI

ABSTRACT

In plasma physics, Two-stream instability is a typical microscopic instability which is important in plasma kinetics. The approaches to solve electron distribution function directly in phase space give us better understanding than such particle methods as PIC, but it takes much longer CPU time. We study an acceleration of the calculation for Vlasov-Poisson equation by using GPU hardware (nVIDIA GeForce8800 GTS (G92 Core)). The Vlasov equation is an advection equation and we have 85 times acceleration than the case of CPU calculation and it corresponds to 120 GFLOPS. However one-dimensional Poisson equation and the integration calculation in velocity space are inefficient because of less thread numbers. The overall acceleration of about 20 times that of CPU is achieved for the calculation of two-stream instability

Keywords: plasma, Two-stream instability, GPU, CUDA

1. 緒 言

プラズマの 2 流体不安定性は位相空間における不安定性としてよく知られていて、プラズマ運動論を理解する上で重要な問題である。PIC 法のように粒子モデルで計算する方法と比較して、電子の速度分布関数を直接解く方法はノイズが無く、速度分布の末端まで高精度で解ける半面、多次元では消費メモリや計算時間などの問題点がある。最近では CIP 法[1]による速度分布の高精度計算が行われ、プラズマの微視的振舞いの詳細[2]が調べられてきた。

一方、CPU を多数用いたスパコンによる高速計算に代わり、GPU(Graphics Processing Unit)の優れた高速演算性能を利用し、グラフィックス処理だけでなく、汎用計算に用いる試みが GPGPU(General-Purpose computation on GPU)として注目されている。GPU は CPU に比べて広いメモリバンド幅と高速な浮動小数点演算器 SP(Streaming Processor)を備えているハードウェアである。また、これまではグラフィックス機能に特化した言語にてプログラムを記述する必要があったが、昨今、CUDA[3]と呼ばれる、既存の CPU でのプログラミングモデルへの親和性が高いフレームワークがリリースされ、GPU を演算デバイスとして利用しやすい環境が整ってきた。

本論文では、プラズマ物理に重要な位相空間 2 流体不安定性を GPU 上で高速計算する方法と、得られた計算速度について述べる。

2. 2 流体不安定性

非常に高温で無衝突近似の成り立つ場合、プラズマの静電モードは、位相空間の Vlasov 方程式と静電場に対する Poisson 方程式に支配される。これらの規格化された方程式は式(1),(2)となる。最も簡単な系として、位相空間は 1 次元の速度空間 v と 1 次元の実空間 x に対して

$$\frac{\partial f}{\partial t} + v \frac{\partial f}{\partial x} - E \frac{\partial f}{\partial v} = 0 \quad (1)$$

$$\frac{\partial^2 \phi}{\partial x^2} = \rho - 1, \quad E = -\frac{\partial \phi}{\partial x} \quad (2)$$

$$\rho = \int_{-\infty}^{\infty} f dv \quad (3)$$

となる。 f は電子の速度分布密度、 E は静電場、 ρ は電子の電荷密度、 ϕ はスカラーポテンシャルを表す。CPU、GPU を問わず、(1)~(3)式の離散化は同じである。

3. GPU 上での計算方法

3.1 移流方程式の計算

高精度計算を目的とするため、双曲型の移流方程式である(1)式には CIP 法を用いる。最も高精度な離散化である TYPE-C[4]を用いるため、従属変数は各空間次元方向の微分に 2 階微分も加えた f , f_x , f_v , f_{xv} を従属変数とする。これらに対する方程式は(1)を x 微分、 v 微分、 xv 微分した式である。2 次元の位相空間 xv の格子

点上に定義された従属変数に対し、CIP 法の TYPE-C は上流点を 3 次精度で補間して次の時刻の値とするセミ・ラグランジュ解法である。この補間プロセスにおいて隣接点の参照が必要となる。

本研究で用いた nVIDIA の G80 および G92 系の GPU 上のハードウェアについて Fig.1 に示す。従属変数を格

Fig. 1 Architecture of G80&G92-GPU

納するために使用するメモリは通常ビデオメモリと呼ばれ、CUDA では Global memory(使用したハードウェアは 512MB 搭載)と呼ばれていて、全ての SP からアクセスできる共有メモリである。また、8 個の SP から構成されるマルチプロセッサは Shared memory(容量 16KByte)と呼ばれるレジスタ並の高速メモリを共有で使うことができる。移流方程式を CIP 法で解く際、計算領域を CUDA の仕様で言う block に分割し、その内部に thread を発生させる。この thread は位相空間の 1 格子点またはせいぜい数点の計算を行うだけで、計算全体としては 1 ステップ当たり数 1000 スレッドとなる。

1 つの thread の実行内容は、Global memory からロードした時刻 n ステップの従属変数のデータを一時的に Shared memory に格納しておく。thread の実行は独立であり、Global memory か Shared memory のデータ以外は他の thread の計算内容にはアクセスできない。Global memory へのアクセスは Shared memory へのアクセスと比べると数倍以上遅く、CIP 法の TYPE-C 補間で参照する隣接点が Shared memory 上にあれば、隣接点のデータを Global memory からロードする必要がなくなり高速計算が可能になる。しかし Shared memory の容量が 16kByte しかないため、使いまわすことで有効に利用する必要がある。

3.2 Poisson 方程式の計算

高精度な移流計算とバランスするように、Poisson 方程式もスカラーポテンシャル ϕ とその x 微分を従属変数として導入し、以下の空間 4 次精度の離散化式[5][6]を

$$\frac{\partial^2 \phi}{\partial x^2} = \frac{2}{\Delta x^2} (\phi_{i+1} - 2\phi_i + \phi_{i-1}) - \frac{1}{2\Delta x} (\phi_{x,i+1} - \phi_{x,i-1}) \quad (4)$$

$$= \rho - 1$$

$$\frac{\partial^3 \phi}{\partial x^3} = \frac{15}{2\Delta x^3} (\phi_{i+1} - \phi_{i-1}) - \frac{3}{2\Delta x^2} (\phi_{x,i+1} + 8\phi_{x,i} + \phi_{x,i-1}) \quad (5)$$

$$= \rho_x$$

SOR 法で解く。この計算 thread は完全に並列であるにも関わらず x 方向の依存関係があるので、RED & BLACK 法を用いて計算点を交互に RED だけの格子点の計算後に値を更新し、次に BLACK の格子点のみを計算するというように交互に SOR 法の計算を収束するまで実行する。実行は 1 つの block に格子点数分の ϕ , ϕ_x

を格納する thread を用意する。Poisson 方程式の計算に際しては全 thread が格子点数分しかないので、並列計算の効率は移流計算などに比べると非常に低下してしまう。

現在の GPU は単精度浮動小数点計算しか行えないが、この制限が直接影響するのが行列計算である Poisson 方程式の計算である。逐次緩和法[7]など、Poisson 方程式を直接計算するのではなく、残差方程式を解くことにより計算を小値で行うことにより精度を保つことができるが、ここではマルチグリッド法を用い、粗い格子では残差方程式を解いているため、高速計算と単精度計算利用を両立させることができた。

3.3 分散関数の積分

Poisson 方程式は実空間の方程式であり、右辺項に現れる電子の電荷密度は(3)式の通り速度分布関数を速度空間で積分することによって得られる。従属変数として f, f_x を計算してあるので、格子点間に対しては高次精度

積分が可能になるが、総和計算では f_x はキャンセルされ単純な総和計算になる。 v 方向に block を取ると reduction により総和計算が 2^n 格子点の場合は n 回の計算で済むが、Global memory からのロードが不連続になり転送速度が大幅に低下する。そこで Poisson 方程式と同じ block を取り、 $2^n - 1$ 回の加算処理を行っている。

3.4 高次精度解法による演算

演算の性質を表す一つの指標として、浮動小数点演算回数 / メモリ操作の回数を考える。演算回数は用いる手法・アルゴリズムの問題なので大きくは変更できない。一方、メモリ操作については Shared memory をキャッシュとして効果的に使用することでアクセス頻度や転送量を削減可能である。高速化にあたっては複数回使用される可能性のあるデータを Shared memory に保存し再利用を行い、Global memory へアクセスする回数を減少

させることが重要となる。

今回用いる移流計算や Poisson 方程式の解法のような高次精度解法ではメモリ操作に対する浮動小数点演算回数が他の解法に比べて多く、GPU の性能を引き出すことが可能である。

4. GPU による計算速度実測

用いたグラフィックスカードは ELSA GLADIAC 988 GTS-512MB 1 ボードである。性能の詳細について Table 1 に示し、ホスト側のハードウェア等について Table 2、概観を Fig 2 に示す。

Table 1 Specification of used graphics card

ELSA GLADIAC 988 GTS 512MB		
Processor	Generation	G92
	Number of SPs	128
	Clock of SP[MHz]	1620
	Peak Performance [GFLOPS]	622
Memory	Capacity [MB]	512
	Bandwidth[bit]	256
	Data Transfer Rate[MHz]	1940
	Transfer Rate[GByte/s]	62.1

Table 2 Specification of Host PC

CPU	Xeon E5420 (2.5 GHz) × 2
Memory	4GByte(DDR2 FB-DIMM 667MHz)
Mother Board	Supermicro X7DWA-N
OS	Cent OS 5.1
Compiler	gcc 4.1.2, CUDA SDK 1.1

Fig. 2 Used Machine (2GPU configuration)

位相空間の計算領域 $0 \leq x \leq 4\pi, -5 \leq v \leq 5$ に対し

t=0.0(Initial Condition)

t=47.068

t=94.114

Fig. 3 Temporal Changes of Two-Stream Instability

て、 256×256 の領域に分割する。 x には周期的境界条件とし、 v 方向には固定境界条件としている。まず(1)式

t=141.149

t=188.94

t=188.94

Fig. 3-(continue)

の移流方程式の計算については、4096 という十分な数の thread を発生させるので並列計算の効率としては十分な性能が期待できる。CPU の約 85 倍に高速化でき、120GFLOPS 相当の演算性能が得られた。今回の 2 流体

不安定性の計算量の大半は移流方程式が占めているが、これに分散関数の積分を加えると約 58GFLOPS に低下し、さらに Poisson 方程式の部分を加えると約 30GFLOPS になってしまう。これは積分計算および Poisson 方程式の計算部分が 256 thread しかないことに起因している。実空間が多次元になればこの問題は解消され、並列計算の効率が上がると予想される。

5. 結 言

プラズマ物理において重要な問題である、2 流体不安定性のシミュレーションを GPU 上で CPU よりも 20 倍高速に演算することができた。特に Vlasov 方程式を解く部分にあたる移流項の部分だけ解く場合には 85 倍以上高速に解くことができた。全体の性能を下げる原因は Poisson 方程式求解演算部分において、割り当てることが可能な thread 数が少なく性能が伸びないという点であった。

また、CIP 法のような高次精度解法による本演算の高速化に今回成功したことから、GPU は高次精度解法を処理するのに向いていることを示した。

謝辞

本研究の一部は、日本学術振興会グローバル COE プログラム「計算世界観の深化と発展」及び、科学技術振興機構 CREST「ULP-HPC:次世代テクノロジーのモデル化・最適化による低消費電力ハイパフォーマンスコンピューティング」の支援を受けた。

参 考 文 献

- 1) T.Yabe and T.Aoki : A Universal Solver for Hyperbolic Equations by Cubic-Polynomial Interpolation, Comput. Phys. Commun., Vol.66, No.2-3, pp.219-232, 1991
- 2) T.Utsumi, T.Kunugi, J.Koga : A numerical method for solving the one-dimensional Vlasov-Poisson equation in phase space, Comput. Phys. Commun, Vol.108, No.2-3, pp.159-179, 1998
- 3) nVIDIA 社 : CUDA Programming Guide 1.1, CUDA SDK, Vol.1.1, No, pp, 2007
- 4) T.Aoki : Multi-Dimensional Advection of CIP (Cubic-Interpolated Propagation) Scheme, CFD Journal, vol.4, No.3, pp.279-291, 1995
- 5) T.Aoki : Interpolated differential operator (IDO) Scheme for solving partial differential equations, Comp. Phys. Comm. Vol.102, No.1-3, pp.132-146, 1997
- 6) Sakurai, K., Aoki, T., Lee, W.H., and Kato, K. : Poisson equation solver with fourth-order accuracy by using interpolated differential operator scheme, Computers Math. Applic., vol.43, pp.621-630, 2002
- 7) 李浩, 青木尊之 : Play Station 2 の Emotion Engine による数値計算, HPCS2002, vol.2002, No.4, pp.65, 2002