シミュレーション並列化の実装方法

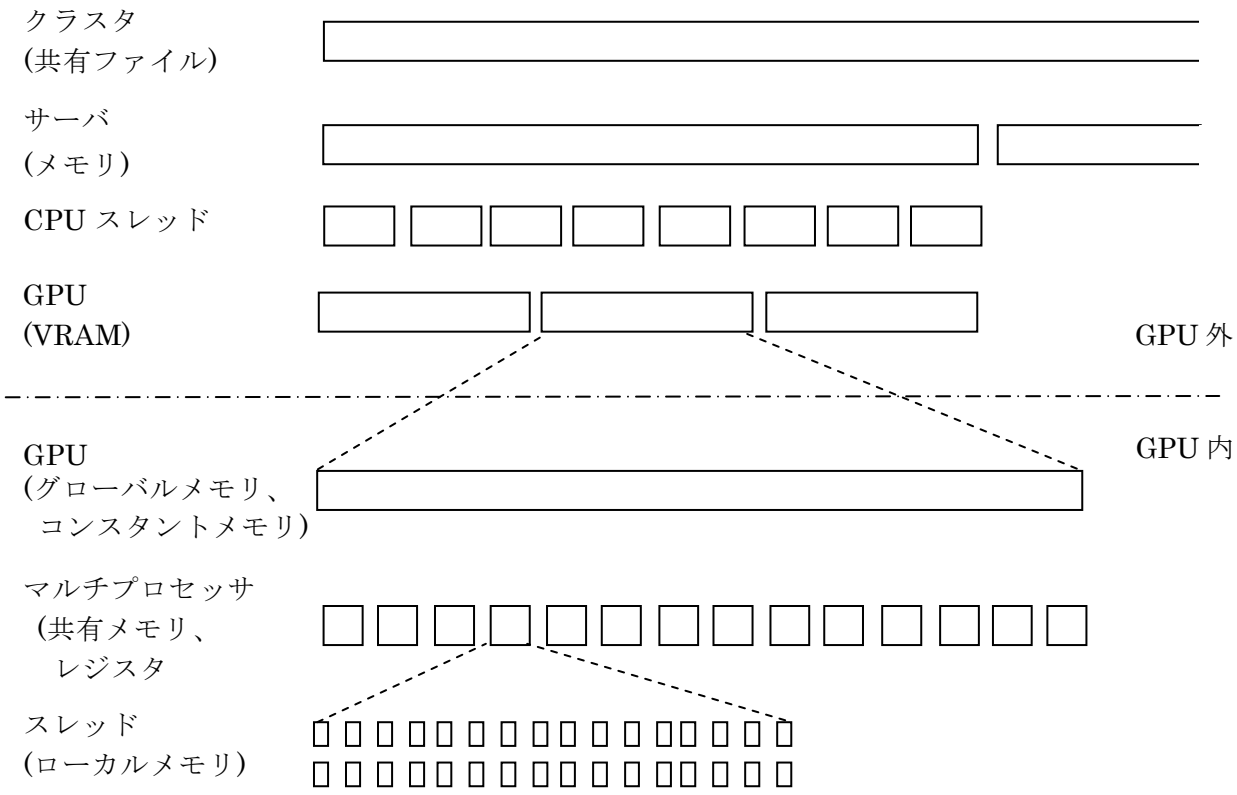
Advanced Algorithm & Systems

1. 概要

シミュレーション並列化の実装方法は、並列化を実現するハードウェアの構成に依存する。
現在のハードウェア構成は多種にわたり、階層的になっている。これらに合わせて並列化を実現するには、各ハードウェアの特性に合わせて適切なソフトウェア実装方法を選択する必要がある。
各ハードウェア構成ごとにソフトウェア実装が変わるため、すべての並列化を統一的に行う方法は無い。並列化の目的に合わせて適切なハードウェアとソフトウェアを選択する必要がある。

2. ハードウェア階層

ハードウェアの階層図を以下に示す。括弧内は記憶媒体である。
GPU 内だけで GPU 外と同程度に複雑である。



3. 複数サーバによる並列化

複数サーバによる並列化は、プログラムを外部へ拡張する形で行われる。

効果として大きいのは速度より計算領域増加である。通信によるオーバーヘッドがあるため、あまり高速化は期待できない。

MPI により並列化を行う。メモリ上のデータを通信で転送する。メインプログラムとデータ領域管理から修正する必要がある。

陰解法の場合は特別な領域分割が必要になる。オーバーラップ領域分割や階層敵領域分割などが用いられる。

4. 複数ＣＰＵスレッドによる並列化

複数 CPU スレッドによる並列化は、プログラムを内部分解する形で行われる。

速度は最大で数倍程度である。メニーコア CPU と呼ばれる多数のスレッドを持つ CPU だと、より速くなる可能性がある。例えば Power 7 なら 8 コア×4 スレッドで実行できる。

MPI か OpenMP により並列化を行う。

MPI は複数サーバと同じ処理になる。通信によるオーバーヘッドが小さくなるため、高速化が期待できる。

OpenMP はループ処理を変えるだけで簡単に並列化できる。ただし効果的に実装しないと速くならない。MPI より遅い可能性もある。

MPI と OpenMP の両方を実装する場合はハイブリッド並列と呼ばれる。

行列処理の一部は単純に分解できない。マルチカラーオーダリング（レッド–ブラック）などの特別な処理により並列化を行う。

5. ＧＰＵによる並列化

GPU による並列化もプログラムを内部分解する形で行われる。

10 倍～100 倍程度の高速化が可能である。ただし高速になる条件が厳しい。

CUDA か OpenCL により並列化を行う。

ループ内を別関数にして呼び出す形で実装する。付加的な処理が幾つか必要になる。直接スレッドまで落とすため、マルチプロセッサはあまり意識する必要がない。ハードウェア構成に合わせた適切な分割実行指示が必要になる。

多数の均等なスレッド分割が必要になる。最低でもスレッド数分（数百）は必要であり、可能なら数万以上の分割が望ましい。これらの処理は均等でないと速くならない。無駄なスレッド占有が発生するためである。マルチプロセッサ内は SIMD であり、同じ処理しか出来ない。

メモリ間のデータ転送処理が必要になる。階層を飛ばしての転送が可能である。適切なアクセス方法で無いと速度が出ない。

サーバのメモリ（GPU のホストメモリ）と GPU 間が遅いため、必要な全データを GPU のグローバルメモリに入れないと早くならない。グローバルメモリは最大で 6G バイト程度であり、これが処理可能なデータ量の上限になる。

できるだけローカルメモリ（最大で 500K バイト程度）にデータを入れて、スレッド内で長い処理を行うと高速化できる。グローバルメモリに対するデータ転送方法も重要である。

コンスタントメモリ、共有メモリ、レジスタを適切に使うと高速化できる。ただしこれらのメモリは小さく、それぞれ数十 k バイト程度である。これらを使わなくても処理可能であるが、アクセス領域が散らばっていると速度が上がらない場合がある。

行列処理の一部は CPU スレッドと同様にマルチカラーオーダリングなどを使う。ただし上記条件を満たさないと高速化できない。