3ZG-1

GPUを用いたエネルギー計算の並列化による ドッキングシミュレーションの高速化

佐々木 崇浩† 宇田川 拓郎† 関嶋 政和†

東京工業大学†

1 はじめに

医薬品の開発において、ターゲットとなるタンパク質に結合するリガンドは、天然資源からの単離やハイスループットスクリーニング (HTS: High Throughput Screening) 等により探索されてきた。しかし、HTS では活性が高いが単位原子当たりの有効性が低い化合物から最適化を始めるために、最終化合物は分子量及び構造の複雑性の増大に伴う物性の悪化が懸念される。この問題に対応するため、活性は低いが単位原子当たりの有効性が高い低分子量化合物に対して、化合物の伸張や結合等の最適化を行い、無駄の少ないコンパクトな最終化合物の作成を可能にするフラグメントベース創薬 (FBDD: Fragment Based Drug Design) への取組みが進められている。タンパク質立体構造情報に基づく薬剤設計では、リガンド結合部位の立体構造解析と静電場の解析（鍵穴構造の解析）から、この部位での相補的構造を有する分子（鍵）をドッキングシミュレーションにより探索する。コンピュータを用いた FBDD では、鍵穴の凹凸それぞれに対して低分子量化合物の結合性を求めるため、膨大な組み合わせのシミュレーションが必要とされる。

近年注目を浴びているのが GPU (Graphics Processing Unit) の利用である。GPU は元来画像処理に用いられてきたが、CUDA (Compute Unified Device Architecture)[1] のような開発環境が整ったこと、CPU に比べ低電力で動作する単純なコアを多数搭載することで、高い並列計算能力とエネルギー効率を有しているため、科学技術計算に用いる試みが盛んに行われている。

そこで本研究では GPU を用いてタンパク質リガンド間ドッキングシミュレーションの高速化を行う。対象にしたのは創薬の研究分野で利用されたドッキングシミュレーションソフトウェアの AutoDock[2] である。

The acceleration of Protein-Ligand docking simulation using GPUs
†Takahiro Sasaki
†Takuro Udagawa
†Masakazu Sekijima
†Tokyo Institute of Technology

2 提案手法

AutoDock は、タンパク質リガンド間のグリッドマップと非結合原子のリストを元に、式 (1) を用いて複合体の結合自由エネルギーを計算し、タンパク質とリガンド間のエネルギーが最小となる構造を求める [2]。

$$V = \sum_{i,j} \left(W_{vdW} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right) + W_{hbound} E(t) \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right) + W_{elec} \frac{q_i q_j}{\epsilon(r_{ij}) r_{ij}} + W_{sol} (S_i V_j + S_j V_i) e^{\left(-\frac{r_{ij}^2}{2\sigma^2}\right)} \right) \tag{1}$$

AutoDock では、最適解を探索するアルゴリズムとして遺伝的アルゴリズム (GA: Genetic Algorithm) を用いている。遺伝的アルゴリズムとはデータを遺伝子として表現し、複数の個体を適応度によって選択し、交叉、突然変異などの遺伝的操作を行いながら最適解を探索するアルゴリズムである。

AutoDock における GA の流れを次に示す。

1. 乱数をによる初期個体の生成
2. 現世代の各個体のエネルギースコアの計算
3. エネルギーをもとに個体の適応度を計算し、次世代へ残す個体の選択
4. 現世代の個体間で交叉を行った後、突然変異の遺伝的操作
5. 最大世代数に達するか最大評価回数に達するまで 2 から 4 まで反復計算
6. 最終世代の個体の中でもっともエネルギーが低い個体を最適解として出力

高速化に先立ち、AutoDock の実行時のプロファイルを解析した結果、AutoDock の GA の計算が実行全体の約 98%を占めていた。その中でも GA のエネルギー計算が実行の約 60%から 70%を占めており、エネルギー計算に GPU を用いて並列に計算し高速化を行った。GPU での実装には NVIDIA が提供している GPGPU 向けの統合開発環境の CUDA を用いた。次に本研究での提案手法を示す。

- **データ構造の書き換え**
  GPU 上でのメモリアクセスが連続になるように入力データの構造を書き換えた。
- **エネルギースコア計算の実装**
  個体ごとのエネルギー計算は各原子について独立している。カーネル関数では各個体ごとにブロックを割り当て、1 つのスレッドが原子のエネルギー計算を担当し、並列計算を行う。
- **コンスタントメモリとシェアードメモリの利用**
  グローバルメモリは一般的にアクセスが非常に遅い。メモリレイテンシを考慮し、3 次元グリッドマップや非結合原子リストなどの入力用のデータにはコンスタントメモリ、入出力用のデータにはシェアードメモリをキャッシュとして使用しさらに高速化を目指した。
- **メモリ転送の隠蔽**
  GPU で計算したエネルギースコアのデータを CPU で用いるには、CPU-GPU 間でメモリ転送を実行する必要がある。メモリ転送を非同期に行い、カーネル関数の実行とメモリ転送をオーバラップさせメモリ転送の隠蔽を行った。

3 実験・評価

既存手法と提案手法のエネルギー計算に要する実行時間について比較実験を行った。一回の GA 実行においてエネルギー計算にかかる時間を gettimeofday 関数を使ってエネルギー計算に要した時間を求めた。実験対象には HIV-2 プロテアーゼにリガンドとしてインジナビルが結合した複合体 (Protein Data Bank ID: 1HSG) を用いた。

本実験は東京工業大学のスーパーコンピュータ TSUBAME2.0 の 1 ノードを用いた。TSUBAME2.0 の各ノードは 6 コアの Intel Xeon X5670 2.93Ghz を 2 ソケット、NVIDIA Tesla M2050 3GPU、54GB の RAM を搭載している。コンパイルには gcc version 4.3.4 および CUDA toolkit version 4.1 を用いた。

遺伝的アルゴリズムのパラメータは交叉率が 0.2、突然変異率が 0.08、GA の実行回数は 10 回である。個体数が 1024 のときと 16384 のときに実験を行い、GA 中のエネルギー計算の平均実行時間の比較を行った。

結果を表 1 に示す。個体数が 1024、16384 両方の場合において、GPU を用いてエネルギー計算した方が CPU で計算したときと比較して約 2.1 倍の高速化に成功した。

表 1: エネルギー計算の平均実行時間の比較

個体数	既存手法	提案手法	計算速度比
1024	53,689 ms	25,529 ms	2.1 倍
16384	53,696 ms	24,039 ms	2.2 倍

4 結論

本研究では AutoDock の実行時間の多くを占めるエネルギー計算を GPU で行うよう CUDA で実装した。個体数が 1024 のときも 16384 のときも GPU を用いてエネルギー計算した方が CPU で計算した時と比較して 2.1 倍の高速化に成功した。

参考文献

[1] CUDA ”The CUDA Zone” http:///www.nvidia.com/cuda

[2] Morris, Garrett M., Ruth Huey, William Lindstrom, Michel F. Sanner, Richard K. Belew, David S. Goodsell, and Arthur J. Olson. ”AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility.” Journal of computational chemistry 30, no. 16 (2009): 2785-2791.

[3] Huey, Ruth, Garrett M. Morris, Arthur J. Olson, and David S. Goodsell. ”A semiempirical free energy force field with charge‐based desolvation.” Journal of computational chemistry 28, no. 6 (2007): 1145-1152.