2K-8

GPUを用いた階層的クラスタリングの高速化

天岸　篤志†　　小柳　滋†
†立命館大学　情報理工学部

1　はじめに

画像処理に使われるGPUを，汎用計算用途に使用するGPGPUに注目が集まっている．CPUと比べ高い演算能力を有し，様々な分野やスケールでの利用が進められている．NVIDIA社は，CUDAと呼ばれるコンパイラやライブラリなどから構成されているGPU向けのC言語の統合開発環境を提供している．

階層的クラスタリングは，多次元空間での類似したオブジェクトのクラスタを決定するために使用される一般的な方法であり，N個のオブジェクトがある場合，$O(N^2logN)$の計算量となるため高速化が望まれている．

本稿では，CUDA/GPUで実装した階層的クラスタリングの並列実行による高速化を目標としている．C言語/Intel Core i5 CPUで実装した逐次実行と比較して処理速度を評価する．

2　階層的クラスタリング

階層的クラスタリングは，以下のようなアルゴリズムである．オブジェクトの総数をn，望むクラスタ数を$k(\leq n)$とする．

1. 各オブジェクトoをただ一つ含むクラスタの集合Cを作る；
2. このとき$|C| = n$である；
3. 条件$|C| > k$が成り立つ間，以下の手順4，5を繰り返す
4. Cの中から互いに最も類似した二つのクラスタc_1，c_2を選んで，Cから削除する；
5. 新たに$c_1 \cup c_2$からなるクラスタc_3を作り，Cに追加する；

上記のステップ3から5を1回実行するたびに$|C|$が1だけ減少するので，上記のアルゴリズムは停止する．

階層的クラスタリングの結果からクラスタを結合していく過程を表す木をデンドログラム(樹形図)という．デンドログラムの例を図1に示す．横軸にクラスタを，縦軸に類似度をとると，類似度が大きいクラスタから結合が始まる．したがって，後から結合されるクラスタ同士は類似度が小さくなる傾向にある．

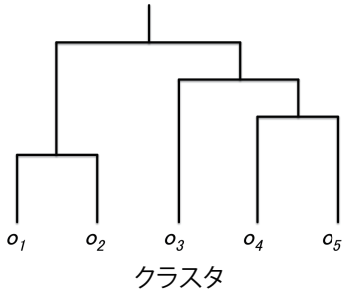

図1: デンドログラム

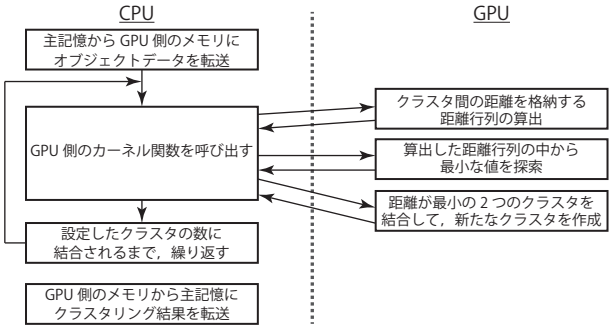

図2: CUDAを用いた階層的クラスタリング

階層的クラスタリングの利点は，近傍値を持つオブジェクトが同じクラスタになり易く，非階層的クラスタリングで用いる初期値がないため結果が初期値に依存することがないことが挙げられる．しかしながら，N個のオブジェクトがある場合，$O(N^2logN)$の計算量となり実行時間が長くなるという欠点がある．

3　実装と評価

3.1　実装

CUDA/GPUで実装する階層的クラスタリングの処理の流れを図2に示す．オブジェクトのデータを主記憶からGPU上のメモリに転送する．GPU側には，距離行列の算出処理，最小値の探索処理，クラスタの結合処理のカーネル関数を実装する．CPU側からGPU側のカーネル関数を呼び出して実行し，クラスタの結合を繰り返して最終的に1つのクラスタに結合されるまで繰り返す．結合されたクラスタのデータをGPU上のメモリから主記憶に転送して，クラスタリングの完了となる．

距離行列の算出処理と最小値の探索処理は，オブジェ

Acceleration of Hierarchical Clustering by GPU
†Atsushi AMAGISHI　†Shigeru OYANAGI
†College of Information Science and Engineering,Ritsumeikan University

表 1: 評価環境

CPU	コア数	動作周波数
Intel Core i5-2500	4	3.30GHz
GPU	CUDA コア数	動作周波数
NVIDIA GeForce 580GTX	512	1.6GHz

クト数の増加につれて計算量も増加する．どちらの処理もメモリアクセス数がとても多くグローバルメモリではレイテンシが大きいため，高速のシェアードメモリを使用してレイテンシを軽減する．

3.1.1 距離行列の算出

オブジェクト数 N の場合 $(N \times 2)^2$ 距離行列を確保する．距離行列の要素数と同じだけスレッドを用いて，各オブジェクト間の距離を並列に計算する．オブジェクトのデータをシェアードメモリに転送しておくことで，メモリのアクセス時間を短縮させる．

3.1.2 最小値の探索

距離行列から配列のリダクションを用いて最小値を見つけ出す．配列のリダクションは，オブジェクト数の増加につれて計算回数も増加するため，シェアードメモリを活用して処理時間を短縮する．バンクコンフリクトを回避をし，またループ処理を展開して分岐処理の増加の原因となる for ループを無くすように実装する．

3.2 評価

CPU と GPU での速度を比較するため，C 言語で階層的クラスタリングを実装した．今回の評価では，乱数を用いて 2 次元座標を持つオブジェクトを生成したデータセットを使用した．また，プログラムの実行時間の計測に使用した評価環境を表 1 に示す．

図 3 は，プログラムの実行時間を示したグラフである．2048 個のデータセットを用いたとき，CUDA 実装が 5056.23 (msec)，C 言語実装が 18928.29 (msec) となり，CUDA 実装が 3.7 倍の高速化を達成した．

図 4 と図 5 は，それぞれ距離行列の算出処理と最小値の探索処理における実行時間を示したグラフである．512 個のデータセットを用いたとき，距離行列の算出処理では，CUDA 実装が 67.40(msec)，C 言語実装が 321.15(msec) となり，CUDAU 実装が 4.8 倍の高速化を達成した．最小値の探索処理では，CUDA 実装が 30.38(msec)，C 言語実装が 99.20(msec) となり，CUDA 実装が 3.3 倍の高速化を達成した．

4 おわりに

本研究では，階層的クラスタリングを CUDA/GPU で実装し，C 言語/Intel Core i5 CPU で実装したプログラムの実行時間の比較を行った．2048 個の 2 次元座標のデータセットを用いて実験した結果，処理全体で 3.7 倍の高速化を達成した．また，ボトルネックとなる処理では，512 個の 2 次元座標のデータセットを用いて実験した結果，距離行列の算出処理で 4.8 倍，最小値の探索処理で 3.3 倍の高速化に成功した．

今後，GPU 側では並列化できない処理を CPU 側で処理をするように改善し，CPU と GPU のハイブリッド実装を行うことで更に実行時間の短縮が可能に成ると考える．

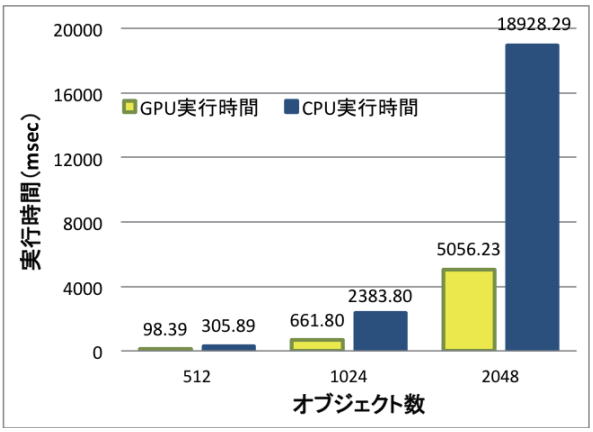

図 3: プログラムの実行時間

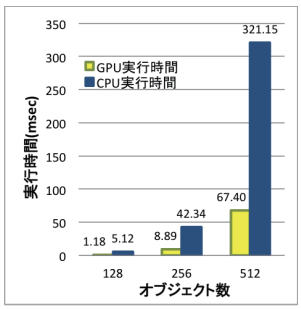

図 4: 距離行列の算出処理の実行時間

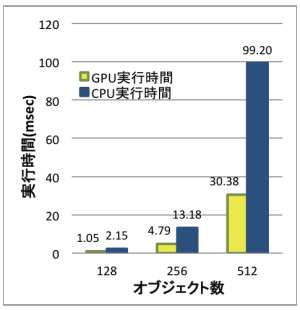

図 5: 最小値の探索処理の実行時間

参考文献

[1] NVIDIA CUDA C Programming Guide Ver 4.0 http://developer.download.nvidia.com/compute/cuda/40/toolkit/docs/CUDA C Programming Guide.pdf

[2] Dar-Jen Chang. Mehmed Kantardzic. Ming Ouyang. Hierarchical clustering with CUDA/GPU. 2009 - bioinformatics.louisville.edu

[3] Zhang Q. Zhang Y. Hierarchical clustering of gene expression profiles with graphics hardware acceleration.Pattern Recog- nition Letters 2006. 27:676-81.