3J-01

GPUによる多次元クラスタリングの高速化

白田　拓馬 †　　小柳　滋 †
† 立命館大学　情報理工学部

1　はじめに

本研究では,GPU を本来の目的である画像処理以外に利用する GPGPU の技術を用いて，CLIQUE と呼ばれる多次元クラスタリングを並列化し，高速化を図る．

2　CLIQUE

2.1　概要

CLIQUE[4] とは，一つの属性を 1 次元とし，隣接する区間の連続から構成させるものと考え，異なる次元における各区間で囲まれたグリッドセルに含まれる点の集合を対象としたクラスタリングの一種である．CLIQUE では，点の集合が高密度クラスタを形成するなら，より低次元でも同じ点の集合は高密度クラスタの一部となるという密度に基づくクラスタの単調性を用いたクラスタリングを行う．
特徴として，高次元の部分集合を自動的に検出し，高密度クラスタが必ず含まれ，スケーラビリティを持っているが，高密度かどうかを決める閾値や区間の設定により精度が左右される不正確さを持ち合わせる．

2.2　アルゴリズム

CLIQUE のアルゴリズムを図 1 に示す
CLIQUE は相関規則発見のアプリオリアルゴリズムに構造がよく似ており，特に $(k-1)$ 次元高密度セルから k 次元の候補集合を生成する部分はアプリオリにおける頻出アイテムセットの候補集合を生成する部分と同じ構造である．

3　GPUによるCLIQUE

本章では，GPU を用いた CLIQUE の処理について述べる．

3.1　概要

CLIQUE の処理において，候補集合生成，枝刈りの一部を GPU で行うことで高速化を図る．GPU に実装するにあたり，Tid と呼ばれる情報を各部分集合に持たせる．Tid(Transaction ID) とは，トランザクションにおいて各部分集合が出現した場所の情報である．

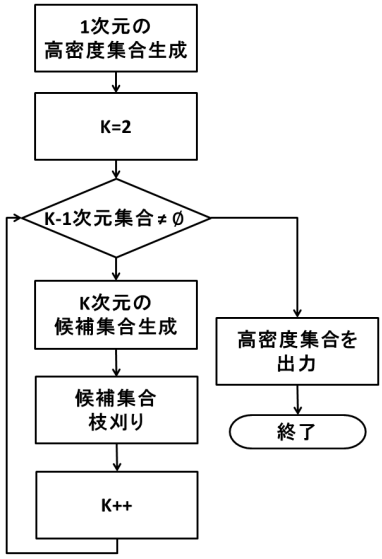

図 1: CLIQUE の流れ

今回はこの Tid を図 2 のように bit に変換し使用する．例えば Tid の属性 $a1$ の場合，トランザクションの属性 a の値 1 が存在するトランザクションの ID は 1,5,6,8 番となり，その部分に対応するビットを 1 とするため，$a1$=10110001 の Tid の bitset が与えられる．

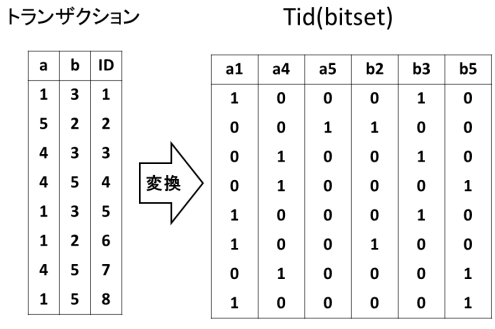

図 2: Tid について

3.2　候補集合生成

k 次元の候補集合生成時，$(k-1)$ 次元の高密度集合を合成することで生成を行うが，合成の際に，$(k-1)$ 次元の高密度集合の Tid を論理積することで k 次元の候

Implementation of High Dimensional Data Clustering by GPU
Takuma Shirota　Shigeru Oyanagi
†College of Information Science and Engineering, Ritsumeikan University

補集合の Tid を算出する．例えば，図 2 の場合に $a1b2$ を生成したとき，$a1b2$ が存在するトランザクションの ID は 6 番目であるが，これは $a1$ と $b2$ の bitset を論理積した結果のビットの位置,0010 0000 からでも判断できることがわかる．Tid は配列に格納されており, 各配列の要素のアクセスおよび，論理積，結果の書き換えを並列処理で行う．

3.3 枝刈り

枝刈りでは高密度かどうかを決めるため, 最小サポートの値以上にセルに点が存在するかどうかで判断される．そのため，部分集合の Tid の 1 の数をカウントし，部分集合が含むトランザクションの数を調べる．

また，Tid は配列に格納されており，各配列の要素ごとにビットカウントが行われるため，各配列の結果をリダクション (総和) する必要がある．リダクションの結果を用いて最小サポート以上であるかどうかを調べ，最小サポート未満である場合に集合を削除し，枝刈りを行う．Tid が格納された配列の各要素へのアクセスおよびビットカウント，リダクションを並列で行う．

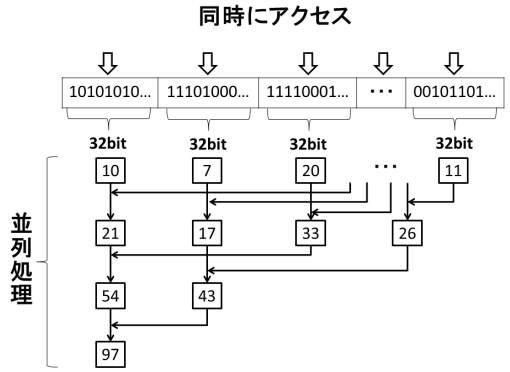

図 3: GPU による枝刈り

3.4 実装

CUDA で実装する GPU による CLIQUE の処理の流れを図 4 に示す.

候補集合生成部では，各 $(k-1)$ 次元部分集合が持つ Tid 配列を GPU に転送し，AND 合成された配列を k 次元部分集合の Tid 配列として格納する. 候補集合削除部でも，部分集合の Tid 配列を GPU に転送するが，結果は結果用の変数を用意し，そこに格納する．また，複数のブロックによる総和計算の場合は変数ではなく配列にする必要があり，一度各ブロックの計算結果を配列に格納し，その配列を再び GPU に転送，計算しなければならない．

3.5 評価

今回，CPU と GPU で実行結果の比較を行うため，C 言語,CUDA で CLIQUE を実装する．今回の評価では乱

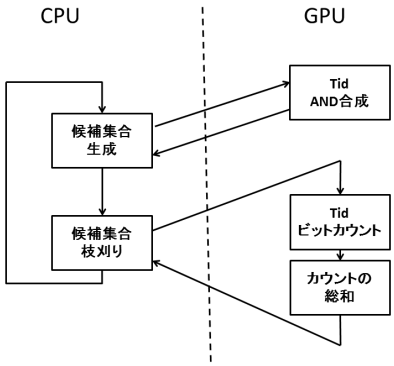

図 4: GPU による CLIQUE の流れ

数を用いてトランザクションをを生成したデータセットを使用．また，プログラムの実行時間の計測に使用する評価環境を表 1 に示す.

実験の手法としては，データ数，次元数と区切り数の増加に伴う実行速度の変化を CPU,GPU 共に確認する．GPU の場合は上記のみならず，CPU,GPU 間におけるデータ転送速度．候補集合生成部と削除部の実行速度を計測し，本手法の有効性を検証する．

表 1: 評価環境

CPU	コア数	動作周波数
Intel Core i7-4770	4	3.4GHz
GPU	CUDA コア数	動作周波数
nVIDIA GeForce GTX 550Ti	192	1.8GHz

4 おわりに

現在 CLIQUE のアルゴリズムを実装している段階であり，実験結果は発表時に紹介する．

5 参考文献

参考文献

[1] 青木尊之 額田彰, ”はじめての CUDA プログラミング”，工学社, 2009

[2] 石川博 新美礼彦 白石陽 横山昌平, ”データマイニングと集合知”，共立出版, 2012

[3] Fan Zhang Yan Zhang Jason Bakos, ”GPApriori:GPU-Accelerated Frequent Itemset Mining”, 2011

[4] R Agrawal J Gehrke D Gunopulos P Raghavan, ”Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications”, 1998