密度に基づくクラスタリングアルゴリズムの マルチコア CPU 上における並列化

Parallel Processing for Density-based Clustering Algorithm on a Muti-core CPU

三崎　浩平
Kohei Misaki
広島市立大学大学院
情報科学研究科
Email: my67022@edu.ipc.hiroshima-cu.ac.jp

田村　慶一
Keiichi Tamura
広島市立大学大学院
情報科学研究科
Email: ktamura@hiroshima-cu.ac.jp

北上　始
Hajime Kitakami
広島市立大学大学院
情報科学研究科
Email: kitakami@hiroshima-cu.ac.jp

Abstract—**Density-based clustering algorithms, which have been well studied in spatial database domains, are based on densities o spatial data. The density-based clustering algorithm can extract spatial clusters with arbitrary shapes. Recently, the sizes and volumes of spatial databases have been increasing because of the popularity of social media. Therefore the speed up for the processing of the density-based clustering algorithms is one of the most important challenges in many different application domains. In this paper, we focus on a density-based clustering algorithm called DBSCAN, which is one of the most basic algorithms. We propose a novel parallelization method for DBSCAN on a multi-core CPU using the spatial partition.**

I.　はじめに

空間データにおいて任意形状のクラスタを取り出す手法として，空間データの密度を基準にした密度に基づくクラスタリングアルゴリズムが提案されている．密度に基づくクラスタリングアルゴリズムでは空間データの近傍に存在する空間データ（近傍データ）数を当該空間データの密度としてとらえ，密度が一定以上大きな空間データを連結していくことでクラスタリングを行う．これは，クラスタ内の空間は密度が高く，クラスタとクラスタ間の空間は密度が低いことの考えに基づいており，人間が空間上の物体を認識するメカニズムを用いた手法となっている．

一般的なクラスタリングアルゴリズムではクラスタの中心からの距離が近いデータがひとつのクラスタとして認識され，円状（高次元では超球状）にまとまりのある集合データがクラスタとして抽出される．一方，密度に基づくクラスタリングアルゴリズムでは，円状ではない任意形状のクラスタを取り出すことができる．例えば，地理情報データで塩害が起こっている地域を取り出す場合，塩害が発生している地域は円状の領域とは限らないため，任意形状で取り出す必要がある．密度に基づくクラスタリングアルゴリズムでは任意形状のクラスタを取り出すことでき，空間データの分析を行う場合に有効な手法のひとつとなっている．

本研究では，密度に基づくクラスタリングの代表的なアルゴリズムのひとつである DBSCAN(Density-Based Spatial Clustering of Applications with Noise）[1, 2]に焦点をあて，マルチコア CPU 上における並列化を試みる．ただし，本研究は，すべてのプロセッサコアが汎用で同一であるようなホモジニアスなマルチコア CPU を研究の対象とする．提案する並列化手法の特徴は次の通りである．

(1)　DBSCAN では空間データの近傍 ε 以内に存在する近傍空間データの数が *MinData* 以上である空間データを空間的に密度が高い空間データとして定義し，そして，空間的に密度が高い空間データを結合していく．よって，近傍 ε 以内に存在する近傍空間データを取り出す処理に時間がかかることが知られており，空間分割を用いて近傍を絞り込むことでこの処理部分の高速化を行う．

(2)　空間分割によるデータ分割並列化手法を用いて問題を分割し，マスタ・ワーカモデルによる動的負荷分散を用いて並列化を行う．DBSCAN では各グリッドで独立にクラスタリングを行うことができ，データ分割の利点を最大限に活用できる．

(3)　複数のグリッドにまたがる空間クラスタはクラスタリング結果をマージすることで抽出する必要がある．ただし，境界領域の密度を正しく判定するために，各グリッドについてグリッドの境界から ε 外に存在する空間データを各グリッドに配置する．

提案する並列化手法で実際に DBSCAN の並列化を行い，ベンチマークデータを用いて評価実験を行った．評価実験の結果，並列化手法の有効性を確認できた．

本論文の構成は次の通りである．第 2 章では，密度に基づくクラスタリングアルゴリズム(DBSCAN)を説明する．第 3 章では，提案手法について述べる．第 4 章で性能評価実験の実験結果を示し，第 5 章で本論文のまとめを行う．

II. DBSCAN

本章では，密度に基づくクラスタリングアルゴリズムのひとつである DBSCAN の諸定義を説明し，DBSCAN のアルゴリズムを述べる．

A. 諸定義

DBSCAN では 2 つの空間データ間の距離を定め，ある空間データ dp からの距離が ε 以内に存在する空間データ dp を ε-近傍 $N_\varepsilon(dp)$と定義する．

定義 1（近傍 $N_\varepsilon(dp)$）：空間データ dp の ε-近傍を $N_\varepsilon(dp)$と表記し，つぎのように定義する．

$$N_\varepsilon(dp)=\{dq \in D | dist(dp, dq) \leq \epsilon\}$$

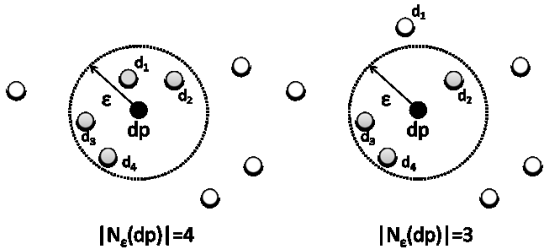

図 1. 定義 1，定義 2 と定義 3 の例

図 1 に定義 1 の例を示す．図 1 左図は，ε 以内に空間データは 4 つ存在し，$|N_\varepsilon(dp)|=4$ となる．一方，図 1 の右では，ε 以内に空間データは 4 つ存在し，$|N_\varepsilon(dp)|=3$ となる．

定義 2（核空間データ，周辺空間データ）：空間データ dp の ε-近傍について，$N_\varepsilon(dp) \geqq$ MinData を満たす空間データ dp を核空間データ，$N_\varepsilon(dp) \leqq MinData$ である空間データを周辺空間データと呼ぶ．

DBSCAN では，核空間データの集合がクラスタの核となり空間クラスタを形成する．図 1 の例だと，$MinData$ =3 とすると，図 1 の左では，空間データは核空間データであり，図 1 の右では，空間データ dp は周辺空間データである．

定義 3（密度的に直接到達可能）：空間データ dp が空間データ dp の ε-近傍であり，$N_\varepsilon(dp) \geqq MinData$ を満たす時，空間データ dp は空間データ dp から ε-密度的に直接到達可能であると表現する．

図 1 の例を使って，例を示す．$MinData$ =4 とすると，図 1 の左では，空間データ dp は核空間データである．つまり，$N_\varepsilon(dp) \geqq MinData$ を満たす．このとき，空間データ d_1，d_2，d_3 と d_4 とは空間データ dp の ε-近傍であり，空間データ dp から ε-密度的に直接到達可能である．

定義 4（密度的に到達可能）：空間データ dp_i が空間データ dp_{i+1} について，空間データ dp_{i+1} が空間データ dp_i から ε-密度的に直接到達可能である，空間データ列$(dp_1, dp_2, \cdots, dp_n)$を考える．この時，空間データ dp_1 と dp_n は，ε-密度的に到達可能であると表現する．

図 2 の例で，ε-密度的に到達可能を説明する．この例では，空間データ列(dp_2, dp_3, dp_4, dp_5)において，空間データ dp_i が空間データ dp_{i+1} について，空間データ dp_{i+1} が空間データ dp_i から ε-密度的に直接到達可能である．よって，空間データ dp_5 は空間データ dp_2 から ε-密度的に直接到達可能である．

定義 5（ε−密度的に接続）：空間データ dp と空間データ dq とが空間データ do と ε-密度的に到達可能であり，空間データ do が $N_\varepsilon(do) \geqq MinData$ を満たす時，空間データ dp と空間データ dq とは ε-密度的に接続していると表現する．

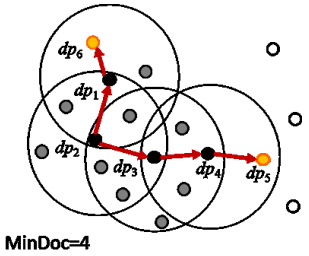

図 2. 定義 4 と定義 5 の例

DBSCAN では近距離に密集している空間データを空間クラスタとして定義している．

定義 6（空間クラスタ）：空間データ集合 D において，空間クラスタ SC は以下の 2 つの条件を満たす部分データ集合である．

(1) 任意の空間データ $dp \in D$ と $dq \in D$ について，空間クラスタ SC に空間データ dp が所属（$dp \in SC$）し，空間データ dq が空間データ dp から ε-密度的に到達可能であれば，空間データ dq は空間クラスタ SC に所属（$dq \in SC$）する．

(2) 空間クラスタ SC に所属する任意の空間データ $dp \in SC$ と $dq \in SC$ は，ε-密度的に接続している．

B. アルゴリズム

空間クラスタは核空間データを併合していくことで抽出する．各空間データ dp_iについて，もし，そのデータがまだどこのクラスタにも所属していなければ，次の処理を行う．空間データ dp_i の ε-近傍を求める．もし，ε-近傍の数が $MinData$ 以上であれば，新しくその空間クラスタ stc_{cid} を作成し，その空間データを空間クラスタ stc_{cid} に所属させる．次に，ε-近傍をキューQ に挿入する．キューQ が空になるまで，Q から空間データ pd を取り出し，次の処理を繰り返す．空間データ pd の ε-近傍を求める．もし，ε-近傍の数が $MinData$ 以上であれば，その空間データ

を空間クラスタ stc_{cid} に加える．次に，ε-近傍の中でクラスタに含まれていないデータのみをキューに加える．

III. 提案手法

本章では，DBSCAN アルゴリズムの空間分割による並列化手法について提案する．

A. データ分割並列化手法

本研究では，データ分割並列化手法を用いて DBSCAN アルゴリズムの並列化を行う．データ分割並列化手法では，パーティション内の処理を行うときに他のパーティションのデータが必要になると効率的に並列化を行うことができない．DBSCAN アルゴリズムでは，空間クラスタは核となる空間データの周辺空間データを結合していくことで作成される．よって，空間クラスタを抽出するときに，遠方にある空間データは不要である．そこで，空間分割によってデータ分割を行うことで，パーティション間の独立性が保たれ，並列化を行うことができる．

空間分割では，空間を複数のグリッドと呼ばれる空間に分割する．空間分割では図 3 左のように領域を均等分割する方式と，図 3 右のように各領域のデータ数が均等となる方式の 2 種類が考えられるが，本研究ではこの 2 種類を実装し，処理性能差を検証する．

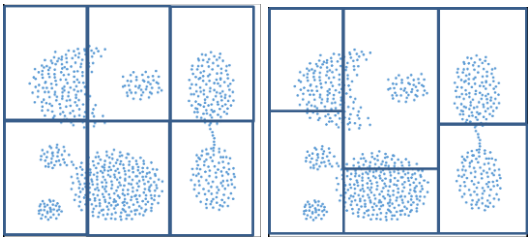

図 3. 領域均等分割と個数均等分割

ただし，各グリッド内は，他のグリッドとは独立に DBSCAN を行い，グリッド内のクラスタを求めることができるが，グリッドの境界から距離 ε 以内に存在するデータは核空間データかどうかの判定がグリッド内のデータのみで行うことができない．そこで，距離 ε だけ多めに領域を広げ，各グリッドが重複した空間データを持つようにする．距離 ε だけ多めにデータ領域を広げることで，本来のグリッドの区分内に存在する空間データについて核空間データであるかどうかの判定を他のグリッド内の空間データにアクセスしなくとも判定することができる．

B. 並列化モデル

並列化では，並列化モデルとして，マスタ・ワーカモデルを用いる．一般的なマスタ・ワーカモデルでは，マスタは仕事の分割とワーカの管理を行う．また，マスタが生成したタスクを暇なワーカが自発的に取ってくることで処理を進める．マスタはタスクを分割し，タスクプールで管理する．各ワーカはタスクプールからタスクを順に取ってきて実行する．タスクプールは，プログラム的にはキューの概念で表すことができ，マスタはタスクを分割してタスクキューに詰め（エンキュー），ワーカはタスクキューを参照してタスクがあれば取り出して（デキュー）実行する．最終的に，マスタがキューに詰めるタスクが完全になくなり，またワーカがキューの中のすべてのタスクを完了したら全体の処理が終了する．

C. 処理手順

本節では並列処理の手順を示す．

1) マスタスレッド:

マスタスレッドの処理ステップを次に示す．

(1) マスタスレッドはパラメータとして，データベース *D*，ε と *MinDoc* を受け取る．

(2) データベース *D* を占める空間について各軸を *p* 分割したグリッドを生成する．そして，各グリッドに空間データを分割する．

(3) マスタスレッドはタスクプールを作る．

(4) マスタスレッドは各グリッドに対する局所的な DBSCAN の実行処理をタスクとしてタスクプールにタスクを格納する．

(5) マスタスレッドは *t* 個のワーカスレッドを作成する．

(6) マスタスレッドは各ワーカスレッドが実施したグリッドごとのクラスタリング結果をワーカスレッドから受け取る．

(7) タスクプールが空でなければ，ステップ(6)まで戻る．そうでなければ，ステップ(8)へ．

(8) マスタスレッドはタスクプールを閉じる．

(9) マスタスレッドはグリッド毎に得られた空間クラスタについて，重複する領域を検出し，結合を行う．

(10) ユーザに空間クラスタ集合を返す．

2) ワーカスレッド:

ワーカスレッドの処理ステップを次に示す．

(1) ワーカスレッドはパラメータとして ε と *MinDoc* を受け取る．

(2) ワーカスレッドはタスクプールからタスクを受け取る．

(3) ワーカスレッドはタスクが示すグリッド内に存在するデータをデータ集合として

DBSCAN を実施し空間クラスタを抽出する．

(4) 抽出した空間クラスタをマスタスレッドに送付する．

(5) タスクプールが空でなければ，ステップ(3)まで戻る．そうでなければ，ステップ(6)へ．

(6) ワーカスレッドを終了する．

D. 空間クラスタの結合

グリッド毎に抽出した空間クラスタに対して，ここでは統合する方法について述べる．領域番号および，各次元の分割地点を比べることでどの領域同士が隣り合った関係にあるかどうかを判定できる．また，各分割領域内の空間データは自分の領域よりも距離 ε だけ広く空間データを得ているので，ひとつの領域に存在するクラスタ内の空間データと自身に隣接している領域のクラスタ内の空間データを比較し，どちらのクラスタにも含まれる空間データが存在した場合，そのクラスタをマージすることでクラスタの統合を行う．

IV. 評価実験

評価実験では，データセット R15，Aggregation，pathbased の 3 種類の異なる分布をしているデータセットを人工的に拡張し，それぞれデータ数 60,000 からなるベンチマークデータを用いて実験を行った．実験環境として，4 コアの CPU を搭載した PC（CPU:Intel Xeon E5-1270 v2 3.5GHz/4 コア/8MB，メモリ：32GB DDR-3(1600MHz/ECC/8GB×4)）を用いた．

図 4 に領域均等分割で，空間分割数を 8 としたときのスピードアップの結果を示す．ワーカスレッドの数を 1 から 4 に変化させて得られたスピードアップを示す．ワーカスレッド数が 4 で約 3 倍のスピードアップが得られた．空間分割や統合時については逐次処理で行われているので 3.1 倍程度になったと考えられる．

図 5 に個数均等分割で，空間分割数を 8 としたときのスピードアップの結果を示す．個数均等分割では最大で 3.3 倍ほどのスピードアップが計れた．また，DBSCAN の計算処理部分が個数均等分割の方が早いという結果も得られたため，個数均等分割による各グリッドの計算時間の減少が最終的なスピードアップに影響を与えることにつながったものと考えられる．

V. まとめ

本研究では，密度に基づくクラスタリングアルゴリズムの代表的なひとつである DBSCAN に焦点をあて，マルチコア CPU 上における並列化を行った．並列化では，データ分割並列化手法を用いて空間を空間分割する効果を検証した．空間分割の効果を確認することができたが，逐次処理部分があるため 3 倍程度のスピードアップという結果になった．これからの課題として，処理全体の並列化，マルチコア CPU の特徴を活かした並列化手法の検討があげられる．

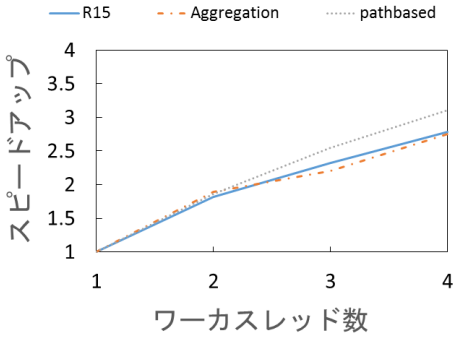

図 4. 領域均等分割時の実験結果

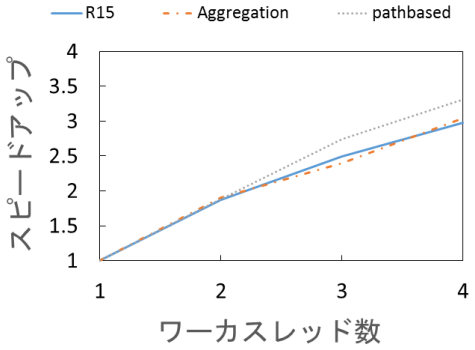

図 5. 個数均等分割時の実験結果

謝辞

本研究の一部は，JSPS 科研費 26330139 と広島市立大学・特定研究費（一般研究，研究課題名「時空間文書ストリーム上におけるバースト領域の抽出手法」）の支援により行われた．

参考文献

[1] Martin Ester, Hans-Peter Kriegel, J¨org Sander, and Xiaowei Xu: A density-based algorithm for discovering clusters in large spatial databases with noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), pp. 226–231, 1996.

[2] J¨org Sander, Martin Ester, Hans-Peter Kriegel, and Xiaowei Xu: Density-based clustering in spatial databases: The algorithm gdbscan and its applications. Data Min. Knowl. Discov., Vol. 2, No. 2, pp. 169–194, June 1998.

問い合わせ先
〒731-3194
広島市安佐南区大塚東三丁目 4 番 1 号
広島市立大学情報科学部知能工学科
三崎　浩平