Vol.2020-MPS-127 No.7
2020/3/2

セルベースのDBSCANのAnytimeアルゴリズムの提案と性能評価

酒井 達弘[1,a)]　田村 慶一[2]　北上 始[3]　竹澤 寿幸[2]

概要：ビッグデータへの注目の高まりにより，データクラスタリング手法の高速化が求められている．本論文では，一定の精度のクラスタリング結果を高速に算出でき，最終的には厳密解を算出可能なセルベースの DBSCAN の Anytime アルゴリズムを提案する．DBSCAN は密度をクラスタリングの基準とした代表的なデータクラスタリング手法の一つである．DBSCAN は計算コストの大きいことが知られており，その高速化手法としてセルベースの DBSCAN が提案されている．セルベースの DBSCAN はデータセット全体をセルごとに分割し，密度をセル単位で考え，セルを結合していくことでクラスタリングを行う．提案するセルベースの DBSCAN の Anytime アルゴリズムは，ランダムに選ばれた一部のセルの結合を行い，高速にクラスタリング結果を算出する．そして，セルの結合とクラスタリング結果の算出を繰り返し行う．提案手法が算出するクラスタリング結果は，処理が進むに連れて高精度となり，最終的には厳密解となる．評価実験の結果，提案手法は既存手法と比較して高精度なクラスタリング結果を高速に算出できることを示した．

キーワード：DBSCAN，クラスタリング，セルベースの DBSCAN，Anytime アルゴリズム

1. はじめに

ビッグデータへの関心の高まりにより，大規模なデータを高速に処理し分析するための技術が求められている．Ester ら [1], [2] によって提案された密度に基づくクラスタリングは，データの密度をクラスタリングの基準とした最も有名なデータクラスタリング手法の一つである．Ester らは密度に基づくクラスタリングの代表的な手法として DBSCAN（Density-based spatial clustering of applications with noise）を提案している．DBSCAN では，データから距離 ϵ 以内の近傍に含まれるデータ数が閾値 $MinPts$ 以上であれば，当該データをコアデータと呼び，コアデータを接続していくことでクラスタリングを行う．DBSCAN には，各データの近傍を求めるための範囲検索とクラスタを形成するための到達可能データ探索という計算コストの大きい二つの処理がある．そこで，数多くの高速化手法が開発されてきた．近年，範囲検索と到達可能データ探索に依存しない手法として，セルベースの DBSCAN[3], [4], [5] が開発されており，大幅な高速化が実現できている．

セルベースの DBSCAN は 2 次元の場合において，データセット全体を一辺が長さ $\epsilon/\sqrt{2}$ の小さいセルに分割する．セル内のデータ数が閾値 $MinPts$ 以上である場合，セル内の全てのデータをコアデータとする．この判定方法によって，範囲検索の回数を大幅に削減できる．そして，コアデータを含むセルを結合していくことでクラスタを形成していく．セルを基準にクラスタを形成するため，到達可能データ探索の処理が必要なく，高速化が実現できる．さらに，我々は先行研究 [6] において，セルの結合判定に最小外接矩形とセルの再帰分割を用いることで高速化を行った．

DBSCAN の改良手法として，AnyDBC と呼ばれる DBSCAN の Anytime アルゴリズムが提案されている [7]．Anytime アルゴリズムとは，任意の時刻で停止でき，その時点でのある程度の精度の解を出力でき，再び処理を再開することができるアルゴリズムである．AnyDBC はランダムに選択したいくつかのデータに対して範囲検索を行いクラスタを形成し，その時点でのクラスタリング結果を算出する．そして，再びいくつかのデータに対して範囲検索を行いクラスタを再形成し，処理が進むに連れて精度の

1 島根大学学術研究院理工学系
Institute of Science and Engineering, Academic Assembly, Shimane University, Shimane 690-0823, Japan
2 広島市立大学大学院情報科学研究科
Graduate School of Information Sciences, Hiroshima City University, Hiroshima 731-3194, Japan
3 広島工業大学情報学部
Faculty of Informatics, Hiroshima Institute of Technology, Hiroshima 731-5143, Japan
a) sakai@cis.shimane-u.ac.jp

ⓒ 2020 Information Processing Society of Japan

高いクラスタリング結果を算出していく．しかしながら，AnyDBC はデータ分布やランダムに選択されるデータによって処理時間が大きくなる場合があり，セルベースの DBSCAN に比べて処理時間が安定しない．

そこで本論文では，処理時間が高速かつ安定している新しいセルベースの DBSCAN の Anytime アルゴリズムを提案する．本論文の貢献は以下の通りである．

- 一定の精度のクラスタリング結果を高速に算出でき，最終的には厳密解を算出可能なセルベースの DBSCAN の Anytime アルゴリズムを提案する．提案手法はランダムに選ばれた一部のセルの結合を行い，高速にクラスタリング結果を算出する．そして，セルの結合とクラスタリング結果の算出を繰り返し行う．提案手法が算出するクラスタリング結果は，処理が進むに連れて高精度となり，最終的には厳密解となる．
- 評価実験により，提案手法は既存手法と比較して高精度なクラスタリング結果を高速に算出できることを示す．

本論文の構成は以下の通りである．第 2 章では，関連研究を述べる．第 3 章では，DBSCAN とセルベースの DBSCAN について説明する．第 4 章では，提案手法について説明する．第 5 章では，評価実験の結果を示し，第 6 章で本論文をまとめる．

2. 関連研究

データクラスタリングはビッグデータ分析において重要な役割を担っており，その高速化に関する研究が盛んに行われてきた．密度に基づくクラスタリング [1], [2] はデータの密度をクラスタリングの基準としており，代表的な手法の DBSCAN は汎用的に使用されているクラスタリング手法である．しかしながら，DBSCAN には各データの近傍を求めるための範囲検索とクラスタを形成するための到達可能データ探索という計算コストの大きい二つの処理がある．そこで，多くの研究者によって DBSCAN の高速化手法が開発されてきた．DBSCAN の高速化手法は，クラスタリング結果が厳密解となる手法と近似解を許す手法に分かれている．一般的に，近似解を許す手法はパラメータやデータ分布によっては得られるクラスタリング結果が厳密解とは異なる場合があるが，厳密解を返す手法よりも高速化することができる．

DBSCAN の高速化手法の一つとして，セルベースの DBSCAN[3], [4], [5] が提案されている．セルベースの DBSCAN は範囲検索の回数を大幅に削減し，小さく分割されたセルを基準にクラスタを形成するため到達可能データ探索の処理を必要としない．さらに，我々は先行研究 [6] において，セルベースの DBSCAN の最も処理時間を要するセルの結合判定に最小外接矩形とセルの再帰分割を用いることで高速化を行った．しかしながら，セルベースの DBSCAN は全ての処理を終えるまでクラスタリング結果を出力することができない．

そこで，DBSCAN の Anytime アルゴリズムとして AnyDBC が提案されている [7]．AnyDBC はランダムに選択したいくつかのデータに対して範囲検索を行いクラスタを形成し，その時点でのクラスタリング結果を算出する．そして，再びいくつかのデータに対して範囲検索を行いクラスタを再形成し，処理が進むに連れて精度の高いクラスタリング結果を算出していく．しかしながら，AnyDBC はランダムに行う範囲検索で多くのデータをカバーできない場合，範囲検索の回数が多くなるため処理時間が大きくなり，セルベースの DBSCAN に比べて処理時間が安定しない．本論文では，処理時間が高速かつ安定している新しいセルベースの DBSCAN の Anytime アルゴリズムを提案する．DBSCAN の高速化の研究において，セルベースの DBSCAN の Anytime アルゴリズムは提案されていない．

3. 事前準備

本章では，DBSCAN とセルベースの DBSCAN について説明する．

3.1 DBSCAN

本節では，DBSCAN について説明する．d 次元のデータ集合を DT とし，ϵ と $MinPts$ をユーザが与えるパラメータとする．データ $dt_p \in DT$ について，dt_p から距離 ϵ 以内に存在するデータ集合を dt_p の ϵ-近傍と呼び，$N_\epsilon(dt_p)$ と表記する．本研究では，データ間の距離はユークリッド距離と定める．コアデータと ϵ-密度的に到達可能を以下のように定義する．

定義 1 (コアデータ) データ dt_p の ϵ-近傍について，$|N_\epsilon(dt_p)| \geq MinPts$ を満たすとき，dt_p をコアデータと呼ぶ．

定義 2 (ϵ-密度的に到達可能) データ列 $(dt_1, dt_2, \cdots, dt_n)$ について，以下の条件を満たすとき，dt_1 から dt_n へ ϵ-密度的に到達可能であると表現する．

(1) $dt_1, dt_2, \cdots, dt_{n-1}$ がコアデータである．

(2) dt_{i+1} が dt_i の ϵ-近傍に存在 $(dt_{i+1} \in N_\epsilon(dt_i))$ する．

DBSCAN において，密度に基づくクラスタはコアデータから ϵ-密度的に到達可能なデータを再帰的に接続していくことで形成されていく．コアデータではないがコアデータから ϵ-密度的に到達可能なデータをボーダデータと呼ぶ．また，どのコアデータからも ϵ-密度的に到達可能ではないデータをノイズと呼ぶ．密度に基づくクラスタの定義を以下に示す．

定義 3 (密度に基づくクラスタ) データ集合 DT におい

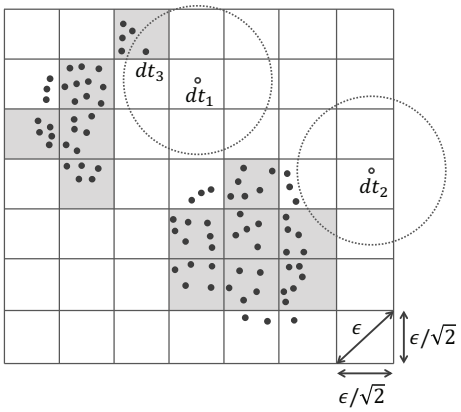

図 1　セルベースの DBSCAN の例
Fig. 1　Example of cell-based DBSCAN.

て，密度に基づくクラスタ DBC は以下の条件を満たす部分データ集合である．

（1）　任意のコアデータ $dt_p \in DBC$ について，DBC に属する全てのデータは dt_p から ϵ-密度的に到達可能である．

（2）　任意のデータ dt_p，$dt_q \in DBC$ について，dt_p と dt_q へ ϵ-密度的に到達可能な $dt_o \in DBC$ が存在する．

3.2　セルベースの DBSCAN

本節では，セルベースの DBSCAN について説明する．セルベースの DBSCAN はセル分割，コアデータ判定，セル結合とボーダデータとノイズ判定の四つの処理から構成される．四つの処理が完了すると，各データはクラスタに属するか，クラスタに属さないノイズとなるかが決定する．

3.2.1　セル分割

最初に，データセット全体を小さいセルに分割し，各データを一つのセルに割り当てる．セルは一辺の長さが同じ超立方体とする．d をデータの次元数とすると，セル一辺の長さは $\epsilon/\sqrt{d}$ となる．セル一辺の長さを $\epsilon/\sqrt{d}$ とすることでセルの対角線の長さは ϵ となる．図 1 にセルベースの DBSCAN の例を示す．図 1 は $d = 2$ であり，$\epsilon/\sqrt{2} \times \epsilon/\sqrt{2}$ のセルに分割されている．

3.2.2　コアデータ判定

次に，各データがコアデータであるかそうでないか判定する．セル内のデータ間の距離は最大でも高々 ϵ であり，セル C_i のデータ数 $|C_i|$ が，$|C_i| \geq MinPts$ を満たすとき，セル C_i 内のデータは全てコアデータと判定できる．反対に，$|C_i| < MinPts$ の場合，セル C_i に属するデータは近隣のセルに存在するデータとの距離計算を行い，コアデータであるか判定する．図 1 の例では，$MinPts = 4$ とすると，グレーで示しているセルにはデータが 4 以上存在し，これらのセルに属するデータは全てコアデータとなる．

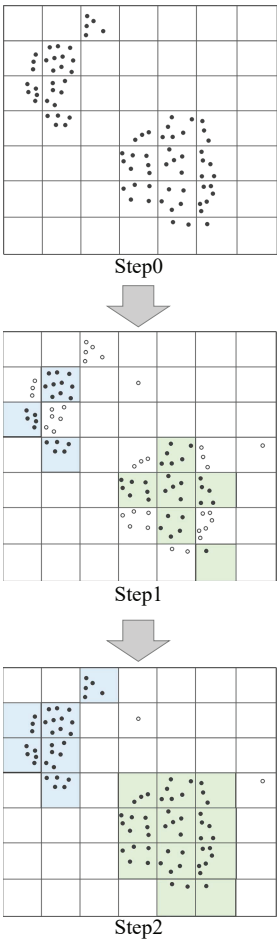

図 2　ランダム選択に基づくセル結合の例
Fig. 2　Example of connecting cells based on random selection.

3.2.3　セル結合

次に，コアデータを含むセルを結合しクラスタを形成していく．二つのセル間に距離 ϵ 以内の任意のコアデータのペアがある場合，二つのセルに含まれる全データは ϵ-密度的に到達可能となり，二つのセルを結合しセル内のデータ集合でクラスタを形成する．セルの結合判定には，先行研究 [6] にて提案された最小外接矩形とセルの再帰分割を用いる．図 1 の例では，左上に密集しているセルに属するデータ集合と，右下に密集しているセルに属するデータ集合の二つのクラスタが形成される．

3.2.4　ボーダデータとノイズ判定

最後に，コアデータではないデータに対してクラスタに属するボーダデータ，もしくはノイズとなるか判定する．判定するデータ dt_i について，ϵ-近傍 $N_\epsilon(dt_i)$ にコアデータが存在する場合，そのデータはボーダデータとなりクラスタに属する．反対に，$N_\epsilon(dt_i)$ にコアデータが存在しない場合，そのデータはクラスタに属さないノイズとなる．図 1 の例では，$N_\epsilon(dt_1)$ に dt_3 が存在しており，dt_1 はボーダデータとなる．反対に，$N_\epsilon(dt_2)$ にはコアデータが存在していないため，dt_2 はノイズとなる．

Algorithm 1 セルベースの DBSCAN の Anytime アルゴリズム

Input: DT, d, ϵ, $MinPts$, p
Output: $ClusterList$，$NoiseList$
1: /*セル分割*/
2: データセット全体を一辺の長さが $\epsilon/\sqrt{d}$ の超立方体に分割し，$CellList$ を作成
3: データ数が 0 個のセルを $CellList$ から削除
4: /*コアデータ判定*/
5: **for** $i = 0$ to $|CellList|$ **do**
6: **if** $|C_i| \geq MinPts$ **then**
7: セル C_i 内のデータをコアデータとする
8: **else**
9: セル C_i 内のデータに対して，近隣のセルに属するデータとの距離計算を行いコアデータ判定を行う
10: **end if**
11: **end for**
12: セルの結合判定を行う必要があるセルのペアリスト $CellPairList$ を計算
13: **for** $i = 0$ to p **do**
14: $ClusterList = \phi$，$NoiseList = \phi$
15: /*ランダム選択に基づくセル結合*/
16: **for** $j = 0$ to $|CellPairList|/p$ **do**
17: $CellPairList$ から結合判定を行っていないセルのペア CP をランダムに選択
18: CP の結合判定を行い，結合できる場合は CP を結合
19: **end for**
20: 結合されたセル内のデータ集合をクラスタとし，$ClusterList$ を作成
21: /*ボーダデータとノイズ判定*/
22: **for** $j = 0$ to $|DT|$ **do**
23: **if** dt_j がクラスタに属していないかつ，$N_\epsilon(dt_j)$ にクラスタに属しているコアデータが存在 **then**
24: $N_\epsilon(dt_j)$ に存在するコアデータが属するクラスタに dt_j を追加
25: **else**
26: dt_j を $NoiseList$ に追加
27: **end if**
28: **end for**
29: 現時点でのクラスタリング結果として，$ClusterList$ と $NoiseList$ を保存
30: **end for**
31: return $ClusterList$，$NoiseList$

4. 提案手法

本章では，提案手法について説明する．

4.1 概要

提案するセルベースの DBSCAN の Anytime アルゴリズムは，セル結合の処理において，ランダムに選ばれた一部のセルの結合を行いクラスタを形成する．そして，結合されたセル以外のセルに属するデータについてボーダデータとノイズ判定を行い，クラスタリング結果を算出する．その後，ランダムに選ばれた一部のセルの結合とクラスタリング結果の算出を繰り返し行っていく．クラスタリング結果は，処理が進むに連れて高精度となり，最終的には厳密解となる．もし，ユーザが処理の途中でクラスタリング結果の出力を求めたとき，そのとき算出されているクラスタリング結果を出力する．

4.2 ランダム選択に基づくセル結合

本節では，提案手法で用いるランダム選択に基づくセル結合について説明する．クラスタリング結果の算出回数を p とする．最初に，セルの結合判定を行う必要があるセルのペアリスト $CellPairList$ を作成する．そして，$CellPairList$ からランダムにセルのペアを選択してセルの結合判定を行い，二つのセルが結合できると判定された場合，セルの結合行う．提案手法はセルの結合判定と結合の処理を $|CellPairList|/p$ 回行う度に，ボーダデータとノイズ判定を行いクラスタリング結果を算出する．p 回目に算出するクラスタリング結果は，$CellPairList$ の全てのセル間の結合判定と結合を終えており，厳密解のクラスタリング結果となる．

図 2 にランダム選択に基づくセル結合の例を示す．図 2 の例は $p = 2$ としており，Step1 では，ランダムに選ばれたいくつかのセルが結合し，二つのクラスタを形成している．提案手法は Step1 の段階で，残されたデータに対してボーダデータとノイズ判定を行い，Step1 の時点でのクラスタリング結果を算出する．Step2 では，全てのセルの結合判定と結合を終えており，最終的な厳密解のクラスタリング結果となる．

4.3 アルゴリズム

本節では，提案手法のアルゴリズムについて説明する．提案手法のアルゴリズムを Algorithm1 に示す．Algorithm1 は入力として，データセット DT，次元数 d，ϵ-近傍の距離 ϵ，閾値 $MinPts$，クラスタリング結果の算出回数 p を入力として受け取り，クラスタリスト $ClusterList$ とノイズリスト $NoiseList$ を出力する．まず，データセット全体を一辺の長さが $\epsilon/\sqrt{d}$ の超立方体に分割してセルリスト $CellList$ を作成し，各データをセルに割り当て，データ数が 0 個のセルは削除する（2〜3 行）．そして，各データのコアデータ判定を行う（5〜11 行）．セルの結合判定を行う必要があるセルのペアリスト $CellPairList$ を作成する（12 行目）．ランダム選択に基づくセル結合を行い，$ClusterList$ を作成する（16〜20 行）．ボーダデータとノイズ判定を行い，$NoiseList$ を作成する（22〜28 行）．現時点でのクラスタリング結果として，$ClusterList$ と $NoiseList$ を保存しておく（29 行）．ランダム選択に基づくセル結合，ボーダデータとノイズ判定とクラスタリング結果の保存を p 回繰り返す（16〜29 行）．もし，処理の途中でユーザが Algorithm1 を停止した場合，保存されている $ClusterList$ と $NoiseList$ を出力する．途中で停止されなかった場合，Algorithm1 は最終的な厳密解の $ClusterList$ と $NoiseList$ を出力する

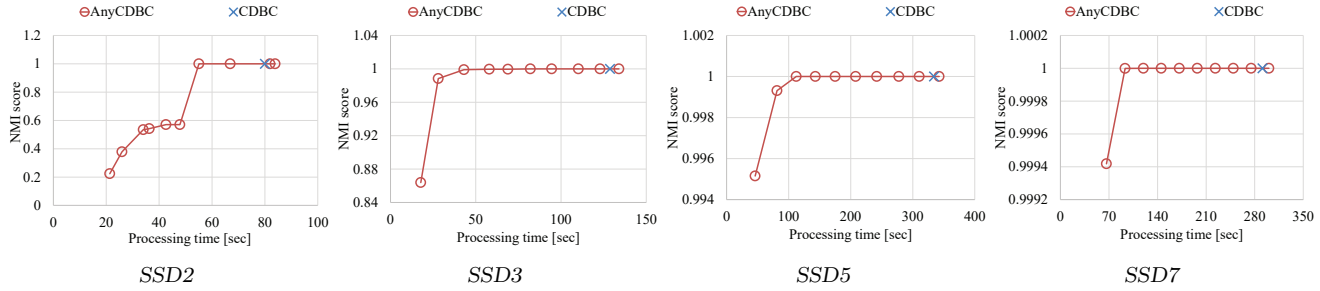

図 3　人工データを使用した実験結果
Fig. 3　Experimental results of synthetic datasets.

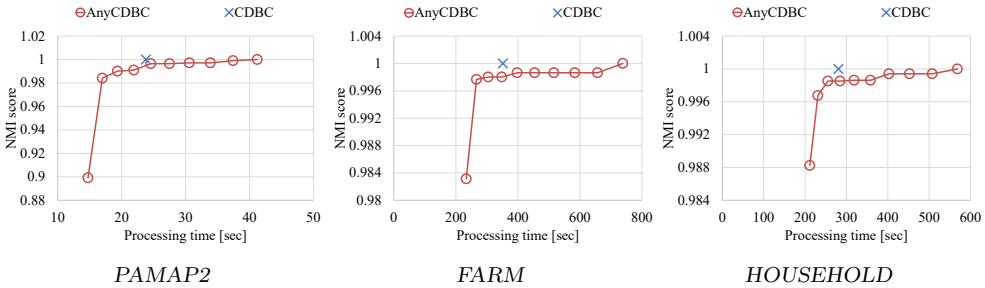

図 4　実データを使用した実験結果
Fig. 4　Experimental results of real datasets.

表 1　データセットの詳細
Table 1　Detail of datasets.

データセット	次元数	データ数
SSD2	2	10,000,000
SSD3	3	10,000,000
SSD5	5	10,000,000
SSD7	7	10,000,000
PAMAP2	4	3,850,505
FARM	5	3,627,086
HOUSEHOLD	7	2,049,280

(31 行).

5.　評価実験

提案手法を評価するために，既存手法との比較実験を行った．本章では，評価実験の結果を示す．

5.1　実験内容とデータセット

評価実験では，提案手法と既存手法の処理時間の比較を行う．途中で算出されるクラスタリング結果については，厳密解との正規化相互情報量（NMI）を評価する．比較手法としては，提案手法（AnyCDBC と表記する），セルの結合判定に最小外接矩形とセルの再帰分割を用いるセルベースの DBSCAN（CDBC と表記する）[6] と DBSCAN の Anytime アルゴリズム（AnyDBC と表記する）[7] を比較する．実験に用いる計算機としては，CPU が Intel Core i7-8700 @3.2 GHz，メモリが 16GB の計算機を使用した．AnyCDBC と CDBC はプログラミング言語として C++を用いて実装し，コンパイラとして g++ 7.4.0，オプションとして-O3 を使用した．AnyDBC は文献 [7] の著者より提供を受けたバイナリファイルを使用した．

データセットは文献 [4] にて開発された Seed Spreader を使用して作成した 2 次元から 7 次元の人工データ（*SSD2*, *SSD3*, *SSD5*, *SSD7*）と文献 [4]，[6] にて使用された実データ（*PAMAP2*, *FARM*, *HOUSEHOLD*）を使用した．Seed Spreader は空間内に密集したデータ集合をいくつか配置し，少量のノイズをランダムに配置することで，クラスタリングのベンチマークに適した分布のデータセットを作成できる．各データセットの次元数とデータ数を表 1 に示す．クラスタリングのパラメータは $MinPts = 100$，$\epsilon = 5,000$ を用いた．また，提案手法のパラメータとして $p = 10$ を用いた．

5.2　実験結果

最初に，人工データを使用した実験結果を図 3 に示す．AnyCDBC は 10 回算出されるクラスタリング結果の NMI の推移を示している．CDBC のクラスタリング結果の算出は 1 回であり，厳密解を返す手法のため NMI は 1.00 となる．図 3 より，*SSD2* は 8 回目まで，その他のデータセットは AnyCDBC は 9 回目までのクラスタリング結果を CDBC よりも高速に算出できている．*SSD2* の 8 回目，その他のデータセットの 9 回目の結果は，既に厳密解と同じクラスタリング結果となっており，AnyCDBC は CDBC

表 2　AnyDBC の処理時間（秒）
Table 2　Processing time of AnyDBC (sec).

データセット	1	2	3	4	5	6	7	8	9	10
SSD2	5900.6	5995.1	92694	201.06	29248	2066.0	902.57	861.77	795.89	792.21
SSD3	44406	44407	50878	47579	51683	81659	41263	61320	40832	42476
SSD5	86.25	86.76	29.31	27.49	584.92	67.03	33.09	84.07	35.72	113.95
SSD7	634.71	636.92	300.05	224.09	350.95	321.90	348.08	461.15	351.06	104.06
PAMAP2	67.48	67.12	66.75	65.78	67.09	71.01	73.52	72.85	65.41	70.20
FARM	207.26	206.33	13855	238.69	126.01	115.11	112.51	145.40	107.74	129.32
HOUSEHOLD	41.07	40.93	40.18	40.34	42.19	39.95	40.44	44.34	41.33	40.57

よりも厳密解を高速に算出できていることが分かる．

次に，実データを使用した実験結果を図 4 に示す．図 4 より，*PAMAP2* と *FARM* は 4 回目まで，*HOUSEHOLD* は 3 回目までのクラスタリング結果を CDBC よりも高速に算出できている．*PAMAP2* と *FARM* の 4 回目，*HOUSEHOLD* の 3 回目のクラスタリング結果の NMI は 0.99 を超えている．実データを使用した場合でも，AnyCDBC は厳密解に近いクラスタリング結果を高速に算出できることを示した．しかしながら，実データを使用した場合，AnyCDBC の最終的な結果は，CDBC の処理時間と比較して大幅に遅くなっている．この原因としては，実データは人工データよりもデータが空間全体にまばらに広がっているためノイズとなるデータ数が多く，クラスタリング結果の算出の度に行われるボーダデータとノイズ判定に多くの処理時間を要したためである．提案手法では，セル結合を一部だけ行いクラスタリング結果を算出していたが，セル結合だけでなくボーダデータとノイズ判定についても，一部のデータのみに対して処理を行い，随時クラスタリング結果を算出するような手法を考える必要がある．

AnyDBC の実験結果を表 2 に示す．表 2 には，AnyDBC の最終的なクラスタリング結果の算出を終えるまでの処理時間を示している．また，AnyDBC は実行毎に処理時間に大幅な差があったため，10 回実行した結果を示している．表 2 より，*HOUSEHOLD* を使用した場合，AnyDBC は AnyCDBC よりも高速であるが，その他のデータセットの処理時間は高速な場合と低速な場合がある．この原因としては，ランダムにデータを選択して行う範囲検索で多くのデータをカバーできない場合，範囲検索の回数が多くなり処理時間を要してしまったためである．AnyCDBC はランダムにセルのペアを選択してセル結合を行うが，結合を行うセルのペアの数は実行毎に等しいため，最終的な処理時間に大きな差は出ない．AnyCDBC は AnyDBC よりも処理時間を高速かつ安定させつつ，DBSCAN の Anytime アルゴリズムを実現できたといえる．

6. まとめ

本論文では，一定の精度を持つクラスタリング結果を高速に算出でき，最終的には厳密解を算出可能なセルベースの DBSCAN の Anytime アルゴリズムを提案した．提案手法はランダムに選ばれた一部のセルの結合を行い，高速にクラスタリング結果を算出する．そして，セルの結合とクラスタリング結果の算出を繰り返し行う．提案手法が算出するクラスタリング結果は，処理が進むに連れて高精度となり，最終的には厳密解となる．

評価実験の結果より，人工データを用いた場合，提案手法は厳密解に近いクラスタリング結果を高速に算出しつつ，最終的な結果も既存のセルベースの DBSCAN とほぼ等しい処理時間で算出できることを示した．実データを用いた場合，提案手法は厳密解に近いクラスタリング結果を高速に算出できたが，最終的な結果は既存のセルベースの DBSCAN の処理時間と比較して大幅に遅くなった．今後の研究としては，セル結合だけでなくボーダデータとノイズ判定についても，一部のデータのみに対して処理を行い，随時クラスタリング結果を算出するような手法を考える必要がある．

参考文献

[1] Ester, M., Kriegel, H.-P., Sander, J. and Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, *In Proc. KDD 1996*, pp. 226–231 (1996).

[2] Sander, J., Ester, M., Kriegel, H.-P. and Xu, X.: Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications, *Data Mining and Knowledge Discovery*, Vol. 2, No. 2, pp. 169–194 (1998).

[3] Gunawan, A.: A faster algorithm for DBSCAN, Master's thesis, Technische University Eindhoven (2013).

[4] Gan, J. and Tao, Y.: DBSCAN Revisited: Mis-Claim, Un-Fixability, and Approximation, *In Proc. SIGMOD 2015*, pp. 519–530 (2015).

[5] Gan, J. and Tao, Y.: On the Hardness and Approximation of Euclidean DBSCAN, *ACM Trans. Database Syst.*, Vol. 42, No. 3, pp. 14:1–14:45 (2017).

[6] 酒井達弘，田村慶一，北上　始，竹澤寿幸：最小外接矩形とセルの再帰分割を用いたセルベースの DBSCAN の高速化，電子情報通信学会論文誌，Vol. J101-D, No. 4 (2018).

[7] Mai, S. T., Assent, I. and Storgaard, M.: AnyDBC: An Efficient Anytime Density-based Clustering Algorithm for Very Large Complex Datasets, *In Proc. KDD 2016*, pp. 1025–1034 (2016).