IADIS International Journal on Computer Science and Information Systems
Vol. 5, No.2, pp. 117-135
ISSN: 1646-3692

A NEW DENSITY-BASED CLUSTERING APPROACH IN GRAPH THEORETIC CONTEXT

Tülin İnkaya. *Industrial Engineering Department, Middle East Technical University, 06531 Ankara, Turkey.*
tulin@ie.metu.edu.tr

Sinan Kayalıgil. *Industrial Engineering Department, Middle East Technical University, 06531 Ankara, Turkey.*
skayali@ie.metu.edu.tr

Nur Evin Özdemirel. *Industrial Engineering Department, Middle East Technical University, 06531 Ankara, Turkey.*
nurevin@ie.metu.edu.tr

ABSTRACT

We consider the clustering problem with arbitrary shapes and different densities both within and between the clusters, where the number of clusters is unknown. We propose a new density-based approach in the graph theory context. The proposed algorithm has three phases. The first phase makes use of graph-based and density-based clustering approaches in order to identify the neighborhood structure of data points. The second phase detects outliers using the local outlier concept. In the third phase, a hiearchical agglomeration is performed to form the final clusters. The algorithm is tested on a number data sets and compared with the well-known clustering algorithms in the literature. Its strengths and limitations are explored in detail.

KEYWORDS

Clustering, density, graph, arbitrary shapes, outlier.

1. INTRODUCTION

Cluster analysis is the organization of a collection of data points in multidimensional space into clusters based on similarity (Jain et al., 1999). Clustering tries to represent the data by relatively fewer clusters and ensures simplicity. For this reason, it is encountered in many research contexts. It is an important and useful step in exploratory data analysis.

Clustering problems have several challenging issues including determination of the number of clusters, handling arbitrary shaped clusters, dealing with density variations within and between clusters, and detection of outliers. Hard (crisp) clustering methods in the literature can be classified into hierarchical, partitional, probabilistic, density-based and graph-based algorithms. In hierarchical methods, clusters are formed either top-down (divisive) or bottom-up (agglomerative). Partitional methods aim at obtaining a single partition of the data set using iterative optimization of a criterion. In probabilistic clustering, data points are assumed to belong to a certain statistical distribution for which parameters are estimated. Density-based clustering considers clusters as dense regions separated by less dense regions. Graph-based approaches consider the problem as a graph in which nodes are the data points and edges represent the similarity between data points. These approaches try to form subgraphs as clusters. In addition to these hard clustering methods, there are also fuzzy clustering algorithms. In fuzzy clustering the membership levels of points in the data set are found instead of assigning a point to a cluster.

Many of the existing clustering algorithms lack a systematic view and they focus on only some of the challenging issues. However, a priori information about the clusters in a data set (e.g. the number of clusters, shape and density of the clusters) is very limited or unavailable. For this reason, it is necessary to consider the clustering problem in a broad framework.

In this work, we propose a new density-based approach. Our approach can handle a combination of the challenging issues for which most of the well-known algorithms have deficiencies. A graph theory context is adopted to address arbitrary shapes and heterogeneous densities in two or higher dimensional space. Unknown number of clusters is assumed. Our algorithm is composed of three phases: neighborhood construction, outlier detection and merging. The first phase uses ideas from both density-based algorithms to handle arbitrary shapes and graphs to deal with varying densities. A neighborhood is constructed for each data point using the proximity and connectivity information. The second phase focuses on outlier detection. Local Outlier Factor (LOF) proposed by Breunig et al. (2000) is revised for the neighborhoods obtained. In the third phase, a hiearchical agglomeration is performed where closures are merged considering the improvement in separation-to-compactness ratio subject to consistency with their neighborhood. In Section 2, we describe the Neighborhood Construction (NC) algorithm. The three-phase heuristic (NOM) is presented in Section 3. Experimental results are presented in Section 4. We conclude in Section 5.

2. NEIGHBORHOOD CONSTRUCTION (NC) ALGORITHM

Widely used neighborhood structures in clustering are based on *k*-nearest neighbors (*k*-NN) or local density of neighbors. *k*-NN is sensitive to parameter *k* for arbitrary shapes and varying densities. In DBSCAN (Ester et al., 1996), neighborhood of a point is defined by a circle with radius ε, and the point is classified as a core point if there are more than *MinPts* points in the circle. Setting ε and *MinPts* is difficult, and different density regions may require different parameter values. DBSCAN can handle arbitrary shaped clusters. However, it is not possible to find clusters with density variations by using *k*-NN or density-based approaches.

NC aims at handling both arbitrary cluster shapes and variations in density. In NC, proximity graphs are used in defining the neighborhood and connectivity of data points. The neighbors of a point are determined using mutual connectivity and density information.

2.1 Notation and Definitions

We use the notation given below in the discussion to follow.

D	set of data points to be clustered (nodes of the graph)
p, q, i, j	indices for data points
d_{pq}	Euclidean distance between points p and q
CC_i, BC_i, PC_i	core, break point, potential candidate sets of point i
CS_i	final candidate set (neighborhood) of point i
Cl_m	set of points in closure (subcluster) m

Clustering of a data set can be interpreted as constructing a disconnected graph where nodes represent the data points and edges connect the data points that are in the same cluster. In the graph theory literature, proximity graphs extract the influence and relevance of nodes in a graph and present proximity information of the nodes. Proximity between any pair of nodes is determined by the distance between the nodes and the existence of other neighboring nodes. We use the Gabriel Graph (GG) in Euclidean space in constructing the neighborhood of data points.

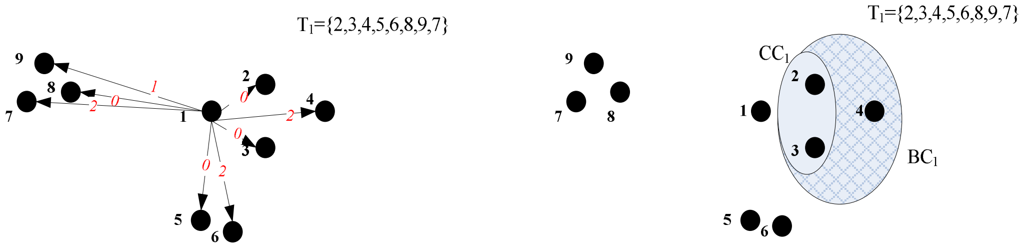
Two nodes p and q are *directly connected* by an edge of the GG if and only if the (hyper)ball having diameter d_{pq} and centered at the midpoint of p and q does not contain any other node of D in its interior. Direct connection makes all connected nodes reachable. Two nodes p and q are *indirectly connected* if the ball with diameter d_{pq} contains at least one other node of D in its interior. This implies that there exists at least one path between the two nodes whose maximum edge length is shorter than d_{pq}. Density between nodes p and q is measured by the number of nodes lying in the ball with diameter d_{pq}.

2.2 Steps of the NC Algorithm

Step 1. Core candidate set construction: In this step, we classify the neighbors of each data point by considering the (direct or indirect) connectivity and density information. For point i, all remaining points in D are listed in non-decreasing order of distance to point i, and the ordered set T_i is formed. The nearest point having an indirect connection to point i is identified as point j. Then, d_{ij} is the first indirect connection distance to point i. Data points having a distance to point i shorter than d_{ij} are directly connected to point i with density 0. We call these data points *core (neighbor) points* of point i and include them in CC_i. Indirect connections to other points will be established via these core points.

Step 2. Break point candidate set construction: Next comes the detection of density change. A cluster is defined as a connected group of patterns of dense neighborhoods (Yousri et al., 2008). Hence, as one moves to the next member of T_i, the density is expected to stay the same or to increase for close neighbors of point i. The first data point in T_i at which the density starts to decrease is identified and called the *break point*. This point may be the sign of a density change (a different cluster). The points that are closer to point i than the break point form BC_i. BC_i is a superset of CC_i, and includes points with indirect connections as well.

Figure 1 shows an example for steps 1 and 2. First, neighbors of data point 1 are ranked and T_1 is formed. Then, density values between point 1 and its neighbors are calculated as 0, 0, 2, 0, 2, 0, 1, 2 in Figure 1(a). The first point subject to indirect connection is point 4, so points 2 and 3 having shorter distance than d_{14} form CC_1. The first density decrease occurs at point 5, which becomes a break point. Points having shorter distance than d_{15} are included in BC_1.



(a) (b)

Figure 1. (a) Density of points in the example data set, (b) Construction of CC$_1$ = {2, 3} and BC$_1$ = {2, 3, 4}

Step 3. Potential candidate set construction: The break point marked in step 2 may indicate either a new density region (a different cluster) or a direction change in the same region (cluster). Premature set wrapping at a break point may cause falling short in defining the neighborhood of a data point. As a remedy, BC$_i$ is extended by checking the connectivity of points. Let k be the first break point of point i. If the intersection of sets BC$_i$ and BC$_k$ is nonempty, then there exists at least one point ensuring an indirect connection between points i and k. Following the points in T$_i$, this check is conducted for every subsequent break point until the first empty intersection of the break point sets is found. Points up to the first empty intersection form PC$_i$.

Let us consider extension of BC$_1$ in Figure 1. Remember that data point 5 is a break point. We check if there exists a direction change or beginning of a new density region at this point. {1} ∪ BC$_1$ and BC$_5$ are compared to check the existence of a point that ensures connectivity between points 1 and 5. BC$_5$ is found as {3, 1, 2, 4} and the intersection is {1, 2, 3, 4} so point 5 is added to PC$_1$. The density increases to 2 for the next member of T$_1$ (point 6). This implies that we are moving along a similar density region, so data point 6 is also in PC$_1$. The next density decrease occurs at data point 8. Again {1} ∪ BC$_1$ and BC$_8$ are compared. This time, the intersection is empty as BC$_8$ is {7, 9}. Hence, extension ends with PC$_1$ = {1, 2, 3, 4, 5, 6}.

Step 4. Candidate set construction: PC$_i$ includes potential neighboring points of data point i. Final decision about a neighboring point is made after a mutual connectivity check. In this operation, PC$_i$ is shrunk to CS$_i$. Let point j be any point in PC$_i$. If point j is in CC$_i$, then CC$_i$ and CC$_j$ are compared for mutuality. If the intersection of these sets is nonempty, points i and j are mutually (nearest) neighbors. So point j is added to CS$_i$. If the intersection is empty, then these points do not share the same neighborhood. Hence, point j and the remaining points in the ordered set PC$_i$ are eliminated from further consideration. If point j is not in CC$_i$, but the intersection of CS$_i$ and CS$_j$ is nonempty, then point j is again added to CS$_i$. These mutuality check operations are conducted for each point until there is no change in any of the CS$_i$ sets.

As an example, let us consider point 7 in Figure 1. When we apply steps 1 through 3, we come up with BC$_7$ = PC$_7$ = {8, 9, 1, 2, 3, 4, 5, 6}. BC$_7$ and PC$_7$ are the same in this case and both sets include points 1-6 from another cluster. The reason is that the density does not show any decrease due to the position of point 7. Even the intersection of CS$_7$ and PC$_1$, the superset of CS$_1$, is empty so mutual connectivity is not satisfied for points 7 and 1. Step 4 eliminates point 1 and the points with longer distance than d_{71} from CS$_7$.

Step 5. Formation of closures (subclusters): Points with common neighbors imply that these points are connected. Thus, closure sets Cl$_m$ are formed by taking the union of CS$_i$ sets that have points in common. Closures formed as such constitute the skeleton of the target clustering solution.

Two main outputs of NC are neighborhood of each point, CS_i, and closures, Cl_m. However, two complications may occur in the neighborhoods constructed. (1) Outlier mixing: If there exist more than one core point for an outlier and if these core points are mutual core neighbors, then outlier mixing occurs. (2) Divided clusters: Because NC lacks a global view of the data set, some local density decreases are taken as different density regions. This causes formation of closures that are smaller than the target clusters.

Time complexity of step 1 of NC is governed by the GG construction and $O(n^3)$ with n nodes. Break point candidate set construction in step 2 takes $O(n^2)$ time. Potential candidate set extension in step 3 has time complexity of $O(n^2)$. Step 4 checks the mutual connectivity of data points with their neighborhoods in $O(n^2)$ time. Step 5 forms closures in $O(n^2)$ time. Since step 4 is repeated until no change occurs in neighborhoods, we cannot determine the overall time complexity of the algorithm, but we can infer that it is at least $O(n^3)$.

3. NEIGHBORHOOD CONSTRUCTION - OUTLIER DETECTION - MERGING (NOM) ALGORITHM

We use the following additional notation in describing NOM.

m, n	indices for clusters
d_{ij}^{GG}	Gabriel Graph (GG) distance between points i and j
lrd_i	local reachability distance for point i
LOF_i	local outlier factor for point i
$i(m)$	point i in cluster m
MST_m	set of edges in the Minimum Spanning Tree (MST) of the points in cluster m
$\mathrm{MST}_{i(m)}$	set of edges in the MST of the points in the neighborhood of point i in cluster m
GG_{ij}	set of edges in the GG of the points that are in the ball centered at the midpoint of points i and j with diameter d_{ij}
NGG_m	set of clusters in the GG neighborhood of cluster m
C_m	set of points in cluster m
sep_{mn}	single link separation between clusters m and n
$comp_{(i)m}$	compactness for the neighborhood of point i in cluster m

Three phases of the NOM algorithm are described below.

Phase 1. Neighborhood construction: This is done with the NC algorithm described in Section 2.

Phase 2. Outlier detection: An outlier is a point that shows abnormal behavior in a data set. In the literature, there are algorithms that extract both clusters and outliers, such as CURE (Guha et al., 1998) and DBSCAN (Ester et al., 1996). However, they specialize in the detection of global outliers, and neither intercluster nor the intracluster density variations are considered. In Breunig et al. (2000), points that are outlying relative to their local neighbors are defined as local outliers. They use a parameter to define the number of points in a neighborhood and compute a Local Outlier Factor (LOF) for each point using this neighborhood. LOF represents the degree of being an outlier based on relative comparison of the average reachability distances of a point and its neighbors.

We are interested in both global and local outliers. Thus, we identify the outliers using a revised version of LOF. Instead of using a fixed parameter to define the size of the neighborhood, we use the neighborhoods constructed in step 1. As the NC algorithm makes use of GG connectivity, resulting neighborhoods can have different sizes and arbitrary shapes. Traditional distance calculation schemes may mislead the density calculation, therefore we consider the GG distance between two points in local reachability calculation. The GG distance takes into account the connectivity between two points. It is the edge with the maximum length in the GG of the points circumscribed by the ball passing through points *i* and *j*, i.e. $d_{ij}^{GG} = \max_{(k,l)\in \mathrm{GG}_{ij}} \{d_{kl}\}$. Then, the revised local reachability density and LOF becomes $lrd_i = \left(\frac{\sum_{j\in \mathrm{CS}_i} d_{ij}^{GG}}{|\mathrm{CS}_i|} \right)^{-1}$ and $LOF_i = \frac{\sum_{j\in \mathrm{CS}_i} \frac{lrd_j}{lrd_i}}{|\mathrm{CS}_i|}$. Given a threshold level, *a*, if $LOF_i > a \max_{j\in \mathrm{CS}_i}\{LOF_j\}$, then point *i* is called a local outlier.

Computational complexity of the outlier detection phase is O(n^3) due to GG construction.

Phase 3. Merging: Hierarchical agglomerative clustering methods construct clusters in stages. Among these CURE (Guha et al., 1998) uses a fixed number of representative points to define the clusters. Agglomeration of a cluster pair is conducted considering the minimum distance between representatives and this is repeated until the given number of clusters is achieved. Although CURE can handle arbitrary shapes, the parameters including the number of representative points, the number of clusters and shrink factor should be set a priori. One of the complications of CURE is handling intracluster and intercluster density variations. CHAMELEON (Karypis et al., 1999) uses *k*-NN to partition the data set. Merging of these partitions depends on the graph connectivity. That is, relative inter-connectivity and relative closeness are calculated between each cluster pair and compared with a given threshold. Like CURE, CHAMELEON can extract arbitrary shaped clusters with different sizes and densities, but faces problems due to density variations within clusters. As we are interested in arbitrary shaped clusters with varying densities, we propose the following procedure for merging subclusters.

At the end of the first phase, we have closures Cl_m obtained from the NC algorithm. After outliers are separated in the second phase, NC closures may consist of divided clusters. As the first two phases take into account density variations in the neighborhood, they depend on the local view. The whole data set is not considered, so there is a lack of global view in the clustering solution. As a remedy, a hierarchical agglomerative procedure is used for merging the neighboring clusters. In order to consider both global and local patterns in the data, improvement in the separation-to-compactness ratio and dispersion of the neighbors are taken into account as the two merging criteria. Clusters subject to merging are determined by using the GG. Two clusters are in the same GG neighborhood if the ball drawn across the nearest two points of a cluster pair does not include any points from other clusters. The following two criteria are then checked for merging.

Criterion 1. Improvement in the separation-to-compactness ratio: We define the potential compactness of a cluster as the most inconsistent edge in the neighborhoods it contains. MSTs are constructed to identify the connections with the minimum total length in a cluster and in its neighborhoods. Then, each edge in the cluster's MST is compared with the edges in the MSTs of the neighborhoods within cluster *m*. Potential compactness of cluster *m* is defined as $pcomp_m = \max\limits_{(i,j)\in \mathrm{MST}_m} \left\{ \frac{d_{ij}}{comp_{i(m)}}, \frac{d_{ij}}{comp_{j(m)}} \right\}$ where compactness value for the neighborhood of point *i* in cluster *m* is $comp_{i(m)} = \max\limits_{(p,q)\in \mathrm{MST}_{i(m)}} \left\{ d_{pq} \right\}$.

If the current cluster had to be divided, the edge that would define the separation would most probably be the most inconsistent edge with its neighborhood identified by $pcomp_m$.

Let the candidate clusters for merging be 1 and 2 where $(i^*, j^*) = \arg\min\limits_{i\in \mathrm{C}_1,\, j\in \mathrm{C}_2} \left\{ d_{ij} \right\}$. Then $d_{i^*j^*}$ is the separation between clusters 1 and 2. Merging them will eliminate the separation $d_{i^*j^*}$ and it will become potential compactness for the merged cluster. A new separation value will emerge between the merged cluster and the cluster nearest to either 1 or 2. We try to find out whether the current separation-to-compactness ratio will improve after the merging. We normalize the separation to account for heterogenity and calculate the current separation-to-compactness ratio as $csep_{12} = \max \left\{ \frac{d_{i^*j^*}}{comp_{i^*(1)}}, \frac{d_{i^*j^*}}{comp_{j^*(2)}} \right\}$ and

$$current\_sc = \frac{csep_{12}}{\max\left\{ pcomp_1, pcomp_2 \right\}}.$$

We consider the lower bound $lb = \min\limits_{\substack{m\in \mathrm{NGG}_1 \\ n\in \mathrm{NGG}_2 \\ m\neq 2, n\neq 1}} \left\{ sep_{1m}, sep_{2n} \right\}$ as the possible separation value after merging. If we merge clusters 1 and 2, the bound on the new separation-to-compactness ratio becomes $new\_sc \geq \frac{lb/d_{i^*j^*}}{csep_{12}}$ where $d_{i^*j^*}$ is used to normalize the lower bound on separation, and $csep_{12}$ becomes the new normalized compactness of the merged cluster.

If *new_sc* is greater than *current_sc*, we conclude that the separation-to-compactness ratio improves after merging. However, this might still be an incorrect signal for merging, especially for the heterogeneous data sets with large distance variations between clusters. Although the ratio seems improving, the new compactness value after merging might be inconsistent with its neighborhood. For this reason, a second check is conducted for the consistency of the neighborhood.

Criterion 2. Heterogenity of edge lengths in the neighborhood: If the candidate clusters for merging satisfy the first criterion, we consider the separation $csep_{12}$ between these two clusters as the potential compactness. To merge, this new edge should be consistent with the neighborhoods of its end points. Hence, merging is performed if this edge does not worsen the dispersion of edge lengths in the neighborhoods, that is

$$csep_{12} \leq \max \left\{ \frac{\max\limits_{(i,j)\in \mathrm{MST}_{i^*(1)}} \left\{ d_{ij} \right\}}{\min\limits_{(i,j)\in \mathrm{MST}_{i^*(1)}} \left\{ d_{ij} \right\}}, \frac{\max\limits_{(i,j)\in \mathrm{MST}_{j^*(2)}} \left\{ d_{ij} \right\}}{\min\limits_{(i,j)\in \mathrm{MST}_{j^*(2)}} \left\{ d_{ij} \right\}} \right\}.$$

Merging continues until none of the cluster pairs satisfy the two merging criteria simultaneously. Therefore, we cannot determine the overall time complexity of this phase.

4. EXPERIMENTAL RESULTS

Performance of NOM is tested on two groups of data sets. Group 1 data sets are taken from the literature (Asuncion and Newman, 2007; Sourina, 2008; Iyigun, 2008) whereas group 2 is a 3-dimensional control group to explore the capabilities of NOM. Group 2 data sets are composed of letters with non-convex shapes (A, E, O, S) and generated using the four factors presented in Table 1. There are 45 and 24 data sets in groups 1 and 2, respectively. Target clusters are either given by the data source or found by visual inspection.

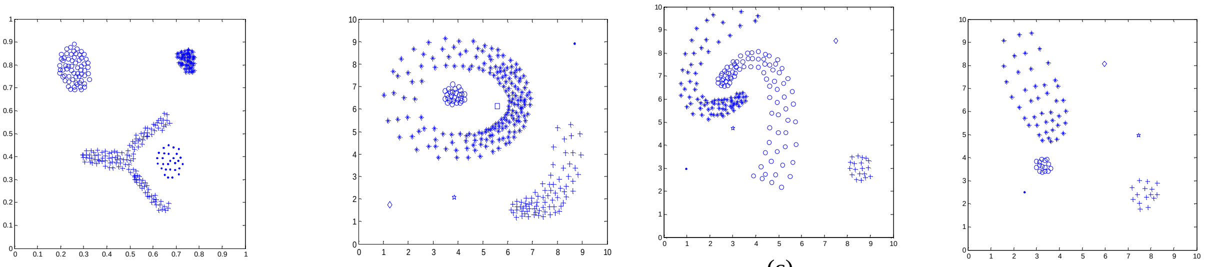
The properties of the data sets are characterized using three measures: the minimum separation-to-compactness ratio (MSCR), the coefficient of variation of the edge lengths in the MST of the whole data set (CV1), and the average of the coefficient of variations of the edge lengths in individual target cluster MSTs (CV2). High values of the MSCR show that even the cluster with the minimum ratio is well-separated from the others, e.g. data_circle. A high coefficient of variation (CV1) for the whole data set indicates well-separated clusters. Large values of CV2 show significant density variations within the clusters, e.g. data-c-cv-nu-n. Group 1 data sets include several types of arbitrary shapes (elongated, curling, ring-shapes, spherical, elliptical, etc.) with density variations. In group 2, in addition to these, proximity of clusters and existence of outliers are explored further. The properties of some sample data sets from two groups are presented in Tables A1 and A2 in the Appendix. The plots of some example data sets are provided in Figures 2 and 3.

Table 1. Factors used in generation of group 2 data sets

		Level 0	Level 1	Level 2
Factors	**Intercluster density difference**	No difference	Clusters having different densities	-
	Intracluster density variation	No variation	Random change	Smooth change
	Intercluster distance	Distant	Close	-
	Outlier	Without outlier	With outlier	-

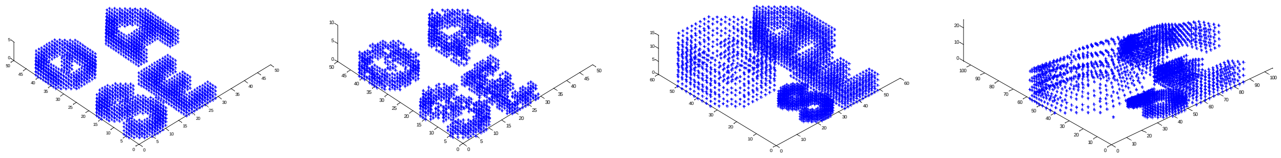
Four performance criteria are used in evaluating the results: the number of clusters, Jaccard index (JI), Rand index (RI) and quasi-Jaccard index (QJI). JI and RI are well-known external cluster validity indices. JI focuses only on the number of point pairs that belong to the same target cluster and assigned to the same cluster whereas RI also considers the number of point pairs that belong to different target clusters and assigned to different clusters. Both of them penalize the divisions and mixes of target clusters. In NC we work on neighborhood construction and we aim to have no mixes from other clusters in the neighborhoods. In order to measure this, we use the relaxed version of JI, QJI, which penalizes only the number of point pairs that belong to the same target cluster and assigned to different clusters. Each measure is calculated for the target clustering solution versus found solutions. The algorithm is coded in Matlab 7.0, and runs are made on a PC with Intel Centrino processor and 512MB RAM.

The performances of NC, outlier detection (OD) and NOM after merging are compared with the results of *k*-means, single-linkage (SL) and DBSCAN approaches. In our comparison *k*-means represents the partitional clustering approach and SL the hierarhical clustering approach. SL also has a graph theoretic view as it has an analogy with MST construction. DBSCAN is selected as a representative of the density-based clustering algorithms. In order to have a fair comparison among these algorithms, *k*-means is run for several values of *k* in the range between 2 and 10% of the points in the data set with increments of 1, and the one with the best JI is used. In the same manner, for DBSCAN, among several *MinPts* settings the one with the best JI is selected for comparison.



(a) (b) (c) (d)

Figure 2. Group 1 data sets (a) train2, (b) data-c-cc-nu-n, (c) data-uc-cc-nu-n, (d) data-c-cv-nu-n



(a) (b) (c) (d)

Figure 3. Group 2 data sets (a) D_0000: no intercluster density difference, no intracluster density variation, distant clusters, no outlier. (b) D_0100: no intercluster density difference, random intracluster density variation, distant clusters, no outlier. (c) D_1010: clusters with intercluster density difference, no intracluster density variation, close clusters, without outlier. (d) D_1211: clusters with intercluster density difference, smooth intracluster density variation, close clusters, with outlier.

The only parameter in NOM, the threshold level *a*, is set to 2 after pilot runs. The details of the results for some sample data sets are given in Tables A3 through A8 in the Appendix. In Tables 2 and 3 the summary of the results for the entire group 1 and group 2 data sets are provided. Clustering results for an example data set is provided in Figure 4. For this data set, JI values with *k*-means, single linkage and DBSCAN algorithms are 0.59, 0.49 and 0.50, respectively. Results of both NC and NOM are superior with respective JI values of 0.98 and 1.

According to Table 2, NOM gives the best average and minimum values of JI and RI over 45 data sets in group 1, as well as the smallest standard deviation. For QJI the best average performance and the smallest standard deviation are achieved by NOM, but *k*-means is better in terms of the minimum. That is, NOM results in clustering solutions close to target clusters. Moreover, the number of cluster mixes is fewer in NOM on the average.

For group 1 data sets, which include arbitrary shapes, intercluster and intracluster density variations, NOM gives the best performance among all the clustering algorithms. Using density-based connectivity through GG, NC is the initial phase for detecting both arbitrary shapes and density changes in the clusters. Outlier detection based on the neighborhoods ensures separation of such points in less dense regions. Merging is performed wherever the separation-to-compactness ratio indicates an increase. The relative evaluation of compactness and separation values according to the neighborhoods in clusters helps handling arbitrary shapes and density differences.

In data sets in which clusters are well-separated and there are ruptures in the intracluster density variations (e.g. data_circle_20_1_5_10 and data_mix_uniform_normal), NOM solution has more clusters than the target solution whereas the clustering solutions obtained by single-linkage and DBSCAN are better. The main reason is the lack of a global view in NOM. In particular, both the NC and the outlier detection phases of NOM have a local view as the decisions are made depending on the information gathered from the neighborhoods. Merging in the third phase tries to bring about a global perspective by checking the improvement at a larger scale, that is, neighborhoods of the clusters instead of points. However, the scale we consider seems to be insufficient to fully realize this.

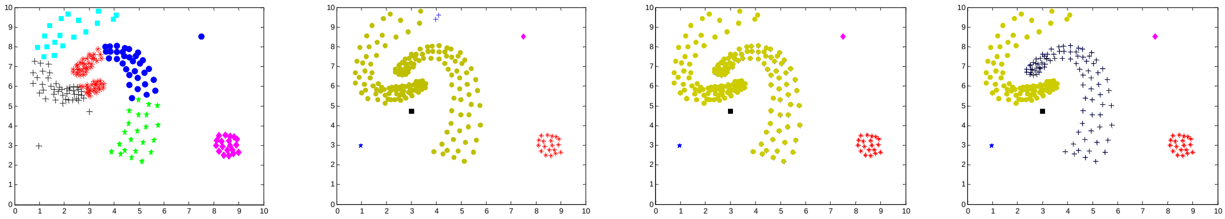
In group 1 experiments target clusters are achieved in 9, 32, 17, 13, 16 and 23 data sets for *k*-means, single linkage, DBSCAN, NC, outlier detection phase of NOM, and NOM, respectively. Note that merging operations in the third phase worsen the performance of NOM in three data sets. In fact these three data sets are the only ones that have JI smaller than 0.80 in NOM. One of them (train3) having the worst performance in JI (0.59), includes noise rather than a few outliers. JI is calculated greater than 0.90 after NC and outlier detection phases and most of the noise is detected as outlier. However, in the merging phase of NOM noise is perceived as a cluster showing similar density properties, so most of these points are merged and clusters made up of noise are formed. We also tested the noise removed version of this data set and NOM was successful in finding the target clusters in this version. As a result, we can infer that NOM is not capable of handling noise. The remaining two data sets that have JI smaller than 0.80 (data_circle_5_10_8_12 and data_circle_3_10_8_12) include intermingled clusters. JI is greater than 0.75 after NC and outlier detection phases. However, the close proximity between the clusters prevents the algorithm from detecting different density regions by the separation-to-compactness ratio, and the clusters are merged in the third phase. Consequently the limitations of NOM are handling data sets with intermingled clusters and noise.

Group 2 is used to explore the main limitations and strengths of NOM further. In this controlled experiment target clusters are achieved in 12, 6, 8, 7 and 7 data sets with single linkage, DBSCAN, NC, outlier detection, and NOM, respectively. *k*-means could not find the target clusters in any of the data sets in group 2, although it seems the best in terms of RI. The letters in group 2 are non-convex, but the shapes are not intertwined. Thus, the center calculation in *k*-means is still useful, and *k*-means shows an average performance in all data sets. As seen from Table 3 NOM is no more the best performer, and DBSCAN and *k*-means have higher JI averages. However, both algorithms find the target clusters in fewer data sets than NOM. Single linkage, having the highest number of successes, does not show good performance in the entire group. DBSCAN having the highest JI achieves the target clusters in only 6 data sets. NOM finds the target clusters in 7 data sets but its JI average is only 0.758. In fact, NOM works well in certain data sets as seen in Table A4 in the Appendix, and performance becomes poor for a certain group. Factorial analysis is conducted to determine the data set properties for which NOM has poor and superior performance.

The effects of the four factors in Table 1 on NOM’s performance (RI) are presented in Figure 5(a). When the density differs among the clusters and the distance between clusters is close (intercluster distance is equal to the distance between the points in the same cluster), RI decreases. The negative effect of smooth density variation is higher than the random intracluster density variation. Note that the existence of outliers does not have a significant effect on the performance of NOM. According to Figure 5(b) the negative effect of the smooth density change increases when the intercluster distance is close. When we exclude the data sets having these properties, the remaining have JI values higher than 0.80. Thus, NOM is capable of handling data sets with intracluster density variations and intercluster density differences when the distance between the clusters is greater than the distance between the closest points in the same cluster. Otherwise, the mixing of clusters seems unavoidable.

To summarize, despite its high performance in JI, RI and QJI, *k*-means cannot find the target clusters. Single linkage performs well when there is no intercluster density difference. DBSCAN mixes outliers and its performance decreases dramatically when there is intracluster density variation (either random change or smooth change) and clusters are close. NOM can handle data sets having arbitrary shapes, intercluster density differences and intracluster density variations, but it fails when clusters are extremely close or when there is noise. To sum up, each clustering approach has its own weaknesses and strengths depending on the characteristics of the approach taken.

Execution times of competing approaches and each phase of NOM are given in Tables A7 and A8 in the Appendix for selected data sets. Execution times of NOM are significantly higher compared to *k*-means, single-linkage and DBSCAN. It spends much time for GG construction, especially for the data sets having a large number of points. Outlier detection takes less time as it requires only one pass of the entire data set. Merging time increases when the number of closures generated by NC (divided clusters) is higher than the number of target clusters (e.g. data_circle). As the dimensionality of the data set increases, the execution times of NOM increase significantly.



(a) (b) (c) (d)

Figure 4. Clustering results for data-uc-cc-nu-n: (a) *k*-means, (b) Single linkage, (c) DBSCAN, (d) NOM

Table 2. Summary results for group 1 data sets

		k-means	Single linkage	DBSCAN	NC	Outline detection	NOM
JI	**average**	0.756	0.937	0.940	0.875	0.875	**0.955**
	std.dev.	0.231	0.163	0.139	0.128	0.137	**0.088**
	min	0.278	0.453	0.504	0.558	0.456	**0.591**
RI	**average**	0.856	0.955	0.963	0.908	0.908	**0.967**
	std.dev.	0.138	0.119	0.095	0.101	0.107	**0.065**
	min	0.580	0.532	0.531	0.659	0.639	**0.648**
QJI	**average**	0.954	0.947	0.972	0.996	0.998	**0.981**
	std.dev.	0.087	0.145	0.097	0.016	0.012	**0.080**
	min	**0.659**	0.460	0.504	0.905	0.916	0.593

Table 3. Summary results for group 2 data sets

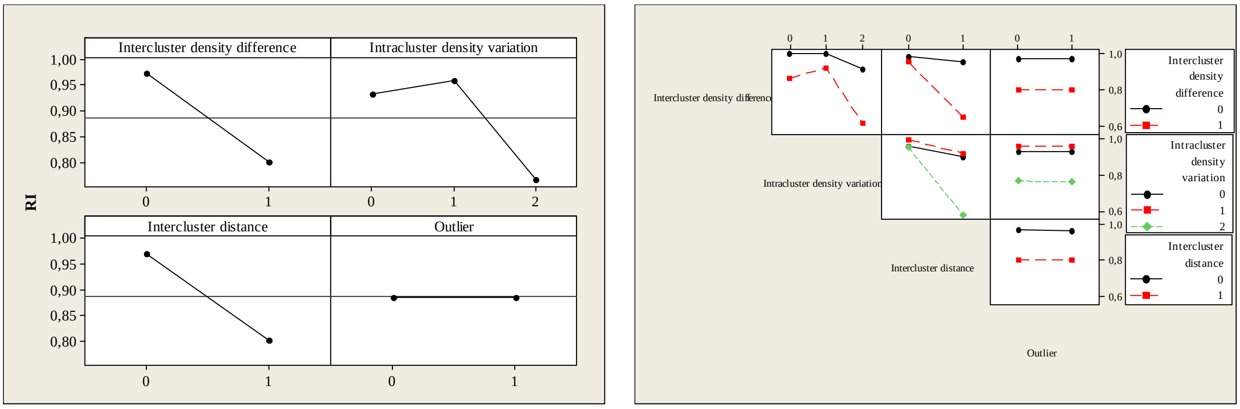
		***k*-means**	**Single-linkage**	**DBSCAN**	**NC**	**Outline detection**	**NOM**
JI	**average**	0.858	0.774	**0.877**	0.740	0.739	0.758
	std.dev.	**0.127**	0.255	0.183	0.257	0.257	0.256
	min	**0.623**	0.328	0.559	0.248	0.248	0.247
RI	**average**	**0.962**	0.887	0.960	0.905	0.891	0.886
	std.dev.	**0.036**	0.153	0.060	0.170	0.166	0.196
	min	**0.895**	0.567	0.843	0.412	0.395	0.285
QJI	**average**	0.938	0.789	**0.939**	0.878	0.878	0.867
	std.dev.	**0.041**	0.238	0.119	0.206	0.206	0.218
	min	**0.876**	0.381	0.674	0.305	0.306	0.259

5. CONCLUSION

NOM is a new density-based clustering algorithm, which uses graph theoretic concepts such as proximity and connectivity as well as density of points in a data set. It has three phases, namely neighborhood construction, outlier detection, and merging of subclusters. It assumes that the number of clusters is unknown. Compared to some other clustering approaches, one of the advantages of NOM is that no parameters need to be set in the neighborhood construction, and only a single parameter (threshold level *a*) is needed in the rest of NOM.

NOM is tested on a number of data sets having various properties and compared with some well-known competing approaches. When the intercluster distances are larger than the intracluster distances, NOM is capable of finding clustering solutions close to the target clusters with arbitrary shapes and different densities. Moreover, NOM can detect the outliers in these data sets although it is not sucessful with noise. Even in the first phase of NOM, the closures obtained after the neighborhood construction are the same as the target clusters for some data sets. Evaluation of compactness and separation measures relative to the neighborhood densities strengthens the capabilities of NOM in handling arbitrary shapes and density variations.

Main limitation of NOM is the lack of collective information from a global perspective. The interrelations among the points are evaluated taking a local view and this results in excessive division of target clusters. More information is needed to handle close clusters having intracluster density variations. Besides stronger mechanisms than the one in phase 3 of NOM can be developed to merge divided clusters. Another complication of NOM is high execution times, but these times can be reduced using efficient coding schemes.



(a) (b)

Figure 5. (a) Main effects of factors on RI, (b) Interaction effects

ACKNOWLEDGEMENT

The authors are grateful to Dr. Cem Iyigun for providing some of the data sets.

REFERENCES

Asuncion, A. and Newman, D.J., 2007. *UCI Machine Learning Repository*. Irvine, CA: University of California, School of Information and Computer Science. Available at: http://www.ics.uci.edu/~mlearn/MLRepository.html [last accessed on 2 May 2009].

Breunig, M.M. et al, 2000. LOF: identifying density-based local outliers. *ACM SIGMOD Record*, Vol.29, No.2, 93-104.

Ester, M. et al, 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Datasets with Noise. *Proceedings of 2nd International Conference on KDDD*, Portland, Oregon, pp 1232-1239.

Guha, S. et al, 1998. CURE: An efficient clustering algorithm for large databases. *Proceedings of the ACM SIGMOD Conference*, Seattle, WA, pp 73-84.

Iyigun, C., 2008. *Probabilistic Distance Clustering*. Ph.D. New Brunswick, New Jersey: Rutgers University.

Jain, A.K. et al., 1999. Data Clustering: A Review. *ACM Computing Surveys*, Vol. 31, No.3, pp 264-323.

Karypis, G. et al, 1999. CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *COMPUTER*, Vol. 32, pp 68-75.

Liu, D. et al., 2008. Effective clustering and boundary detection algorithm based on Delaunay triangulation. *Pattern Recognition Letters*, Vol. 29, No. 9, pp 1261-1273.

Sourina, O., 2008. *Spatial Data Clustering with Boundary Detection.* Available at: http://www3.ntu.edu.sg/home/eosourina/Copy%20of%20ProfessionalService.htm [last accessed on 2 May 2009].

Yousri, N.A. et al, 2008. A Novel Validity Measure for Clusters of Arbitrary Shapes and Densities. *International Conference of Pattern Recognition*, Tampa, USA.

APPENDIX

Table A1. Properties of group 1 data sets

Data set	**# of target clusters**	**# of outliers**	**# of points**	**MSCR**	**CV1**	**CV2**	**min. sep.**	**max. comp.**
data_60	3	0	60	1.50	0.43	0.23	1.46	1.00
data_66	4	0	66	1.50	0.47	0.27	1.27	1.00
data-c-cv-nu-n_v2	3	0	73	1.02	1.04	0.25	0.80	0.78
data-c-cv-nu-n	6	3	76	1.02	1.04	0.25	0.80	0.78
data-c-cv-u-n	5	3	81	2.74	1.13	0.24	1.79	0.65
data-uc-cv-nu-n	6	3	127	0.92	1.04	0.32	0.62	0.67
data-oo_v2	2	0	140	2.52	0.47	0.16	0.46	0.55
data-oo	6	4	144	2.52	1.46	0.16	0.46	0.55
iris	3	0	150	0.35	0.60	0.46	0.22	0.91
data-uc-cc-nu-n_v2	3	0	188	0.80	0.78	0.42	0.54	0.68
data-uc-cc-nu-n	6	3	191	0.80	1.04	0.42	0.54	0.68
data-c-cc-nu-n2_v2	3	0	192	3.31	0.63	0.24	1.82	0.55
data-c-cc-nu-n2	6	3	195	1.72	0.79	0.24	0.95	0.55
dataX_v2	2	0	200	1.15	0.64	0.63	1.04	0.90
dataX	4	2	202	1.15	0.75	0.63	1.04	0.90
data-c-cc-nu-n_v2	3	0	285	1.07	0.56	0.37	0.82	0.77
train2	4	0	287	2.79	1.23	0.27	0.07	0.03
data-c-cc-nu-n	7	4	289	0.60	0.94	0.37	0.46	0.77
train1_v1	5	1	306	3.02	1.28	0.38	0.05	0.03
3d_dataset3	2	0	325	11.87	0.93	0.13	5.94	0.62
train3	36	30	397	0.03	1.26	0.78	0.02	0.74
data_circle	2	0	700	51.94	2.36	0.59	0.71	0.04
data_mix_uniform_normal	2	0	1000	13.52	1.39	0.71	2.12	0.51
data_circle_2_10_2_12	2	0	1200	15.19	0.82	0.61	0.33	0.08
data_circle_5_10_8_12	2	0	1500	0.46	0.61	0.61	0.04	0.09
3d_dataset4	2	0	1523	29.68	0.71	0.13	5.94	0.62
data_circle1	2	0	1890	3.90	0.67	0.61	0.22	0.06
data_circle_1_20_1_15	2	0	2100	14.99	0.68	0.62	0.23	0.08

Table A2. Properties of group 2 data sets

Data set	# of target clusters	# of outliers	# of points	MSCR	CV 1	CV2	min . sep.	max. comp.
D_0001	8	3	2783	3.00	0.28	0.04	3.00	1.00
D_0011	8	3	2783	2.00	0.29	0.04	2.00	1.00
D_0101	8	3	1978	2.12	0.31	0.06	3.00	1.41
D_0111	8	3	1930	1.41	0.34	0.05	2.00	1.41
D_0201	8	3	2783	1.58	0.51	0.39	3.10	2.00
D_0211	8	3	2783	1.02	0.49	0.39	2.00	2.00
D_1001	8	3	2783	1.28	0.31	0.12	2.20	4.33
D_1011	8	3	2783	0.96	0.31	0.12	2.01	4.33
D_1101	8	3	1928	1.44	0.36	0.17	2.20	4.33
D_1111	8	3	1951	0.96	0.33	0.16	2.01	4.33
D_1201	8	3	2783	1.44	0.55	0.45	4.06	6.05
D_1211	8	3	2783	0.98	0.55	0.45	2.12	6.05

Table A3. Performance of clustering algorithms in terms of JI and RI for group 1 data sets

Data set	*k*-means		SL		DBSCAN		NC		OD		NOM	
	JI	RI	JI	RI	JI	RI	JI	RI	JI	RI	JI	RI
data_60	0.79	0.90	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
data_66	0.66	0.83	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
data-c-cv-nu-n_v2	0.61	0.84	1.00	1.00	0.66	0.86	1.00	1.00	1.00	1.00	1.00	1.00
data-c-cv-nu-n	0.59	0.83	1.00	1.00	0.63	0.68	0.95	0.98	1.00	1.00	1.00	1.00
data-c-cv-u-n	0.93	0.97	1.00	1.00	1.00 Δ	1.00 Θ	1.00	1.00	1.00	1.00	1.00	1.00
data-uc-cv-nu-n	0.62	0.83	0.98	0.99	0.98	0.99	0.98	0.99	1.00	1.00	1.00	1.00
data-oo_v2	0.52	0.76	0.89	0.95	0.95	0.98	1.00	1.00	1.00	1.00	1.00	1.00
data-oo	0.49	0.75	0.50	0.53	0.50	0.53	1.00	1.00	1.00	1.00	1.00	1.00
iris	0.70	0.88	0.57	0.78	0.59	0.78	0.86	0.92	0.86	1.00 Θ	1.00	1.00
data-uc-cc-nu-n_v2	0.34	0.73	0.45	0.60	0.59	0.83	1.00	1.00	1.00	1.00	1.00	1.00
data-uc-cc-nu-n	0.59	0.73	0.48	0.62	0.50	0.83	0.98	0.99	1.00	1.00	1.00	1.00
data-c-cc-nu-n2_v2	0.29	0.63	1.00	1.00	1.00	1.00	0.98	0.99	0.98	0.99	1.00	1.00
data-c-cc-nu-n2	0.28	0.64	1.00	1.00	0.99	0.99	0.98	0.99	0.98	0.99	1.00	1.00

dataX_v2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
dataX	0.98	0.99	1.00	1.00	1.00[Δ]	1.00[Θ]	1.00	1.00	1.00	1.00	1.00	1.00
data-c-cc-nu-n_v2	0.80	0.88	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
train2	0.78	0.91	1.00	1.00	1.00	1.00	0.97	0.99	0.96	0.99	1.00	1.00
data-c-cc-nu-n	0.78	0.86	0.99	0.99	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
train1_v1	1.00[ψ]	1.00[Θ]	1.00	1.00	1.00	1.00	0.95	0.98	0.95	0.98	1.00	1.00
3d_dataset3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
train3	0.37	0.79	0.46	0.64	0.97	0.99	0.90	0.97	0.91	0.97	0.59	0.79
data_circle	1.00	1.00	1.00	1.00	1.00	1.00	0.84	0.88	0.84	0.88	1.00	1.00
data_mix_uniform_normal	1.00	1.00	1.00	1.00	1.00[ψ]	1.00[Δ]	0.56	0.83	0.46	0.73	0.84	0.92
data_circle_2_10_2_12	0.93	0.95	1.00	1.00	1.00[Δ]	1.00[Θ]	0.79	0.85	0.79	0.85	0.94	0.96
data_circle_5_10_8_12	0.93	0.96	0.99	0.99	0.99	0.99	0.77	0.87	0.77	0.87	0.77	0.87
3d_dataset4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
data_circle1	0.59	0.62	1.00	1.00	1.00	1.00	0.66	0.69	0.66	0.69	0.95	0.96
data_circle_1_20_1_15	0.61	0.64	1.00	1.00	1.00[Θ]	1.00[Θ]	0.85	0.86	0.85	0.86	0.99	0.99

[ψ] this value is 0.997. [Δ] this value is 0.998. [Θ] this value is 0.999.

Table A4. Performance of clustering algorithms in terms of JI and RI for group 2 data sets

Data set	***k*-means**		**SL**		**DBSCAN**		**NC**		**OD**		**NOM**	
	JI	**RI**	**JI**	**RI**	**JI**	**RI**	**JI**	**RI**	**JI**	**RI**	**JI**	**RI**
D_0001	0.98	0.99	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
D_0011	0.94	0.98	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
D_0101	0.98	0.99	1.00	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00	1.00
D_0111	0.92	0.98	1.00	1.00	0.99	0.99	1.00[Δ]	1.00[ϴ]	1.00[Δ]	1.00[ϴ]	1.00[Δ]	1.00[ϴ]
D_0201	0.90	0.97	1.00	1.00	0.99	0.99	0.78	0.95	0.78	0.95	0.84	0.96
D_0211	0.81	0.95	1.00	1.00	0.94	0.98	0.60	0.87	0.60	0.87	0.63	0.88
D_1001	0.89	0.97	0.67	0.88	0.98	0.99	0.68	0.92	0.68	0.92	0.68	0.92
D_1011	0.66	0.91	0.33	0.57	0.56	0.89	0.37	0.81	0.37	0.81	0.37	0.81
D_1101	0.98	0.99	0.67	0.88	0.98	0.99	0.97	0.99	0.97	0.99	0.97	0.99
D_1111	0.62	0.89	0.33	0.57	0.56	0.84	0.57	0.85	0.57	0.85	0.57	0.85
D_1201	0.85	0.96	0.67	0.88	0.92	0.98	0.62	0.90	0.62	0.90	0.79	0.94
D_1211	0.64	0.90	0.36	0.59	0.58	0.85	0.25	0.39	0.25	0.40	0.25	0.29

[Δ] this value is 0.998. [ϴ] this value is 0.999.

Table A5. Performance of clustering algorithms in terms of the number of clusters for group 1 data sets

Data set	**#TC***	***k*-means** **#C****	**SL** **#C****	**DBSCAN** **#C****	**NC** **#C****	**OD** **#C****	**NOM** **#C****
data_60	3	2	3	3	3	3	3
data_66	4	2	4	4	4	4	4
data-c-cv-nu-n_v2	3	4	3	5	3	3	3
data-c-cv-nu-n	6	5	6	3	4	6	6
data-c-cv-u-n	5	2	5	3	5	5	5
data-uc-cv-nu-n	6	5	5	4	5	6	6
data-oo_v2	2	5	7	3	2	2	2
data-oo	6	7	5	2	6	6	6
iris	2	3	6	3	4	4	2
data-uc-cc-nu-n_v2	3	8	7	8	3	3	3
data-uc-cc-nu-n	6	6	6	5	4	6	6
data-c-cc-nu-n2_v2	3	7	3	3	4	4	3
data-c-cc-nu-n2	6	7	6	4	7	7	6
dataX_v2	2	2	2	2	2	2	2
dataX	4	2	4	3	4	4	4
data-c-cc-nu-n_v2	3	2	3	3	3	3	3
train2	4	3	4	4	6	6	4
data-c-cc-nu-n	7	2	6	4	7	7	7
train1_v1	5	4	5	5	8	8	5
3d_dataset3	2	2	2	2	2	2	2
train3	36	7	10	6	17	20	14
data_circle	2	2	2	2	14	14	2
data_mix_uniform_normal	2	2	2	3	38	39	12

data_circle_2_10_2_12	2	2	2	3	22	23	8
data_circle_5_10_8_12	2	2	7	3	29	30	30
3d_dataset4	2	2	2	2	2	2	2
data_circle1	2	2	2	2	35	36	9
data_circle_1_20_1_15	2	2	2	3	47	49	8

*#TC: the number of target clusters, **#C: the number of clusters found

Table A6. Performance of clustering algorithms in terms of the number of clusters for group 2 data sets

Data set	**#TC***	***k*-means #C****	**SL #C****	**DBSCAN #C****	**NC #C****	**OD #C****	**NOM #C****
D_0001	8	4	8	6	8	8	8
D_0011	8	4	8	6	8	8	8
D_0101	8	4	8	6	8	8	8
D_0111	8	4	8	6	7	7	7
D_0201	8	4	8	6	67	71	19
D_0211	8	4	8	6	66	71	18
D_1001	8	5	6	4	15	16	16
D_1011	8	5	10	10	15	16	16
D_1101	8	4	6	4	9	9	9
D_1111	8	5	10	18	14	18	18
D_1201	8	4	10	10	44	48	15
D_1211	8	5	10	10	40	45	17

*#TC: the number of target clusters, **#C: the number of clusters found

Table A7. Performance of clustering algorithms in terms of time (in seconds) for group 1 data sets

Data set	***k*-means**	**SL**	**DBSCAN**	**NC**	**OD**	**NOM***
data_60	0.21	0.52	0.07	1.24	2.23	3.74
data_66	0.07	0.38	0.03	1.51	1.08	2.65
data-c-cv-nu-n_v2	0.19	0.39	0.06	2.16	1.71	3.97
data-c-cv-nu-n	0.05	0.38	0.04	2.36	1.91	4.37
data-c-cv-u-n	0.11	0.38	0.24	2.77	3.37	6.30
data-uc-cv-nu-n	0.07	0.42	0.13	10.71	7.44	18.34
data-oo_v2	0.73	0.43	1.95	14.08	13.72	28.04
data-oo	0.11	0.43	0.23	15.15	14.28	29.67
iris	2.19	0.57	7.45	18.21	6.24	25.02
data-uc-cc-nu-n_v2	0.09	0.48	0.13	34.27	9.30	43.81
data-uc-cc-nu-n	0.10	0.48	0.12	35.71	9.37	45.33
data-c-cc-nu-n2_v2	0.32	0.49	0.38	36.14	16.69	53.49
data-c-cc-nu-n2	0.11	0.49	0.19	37.72	17.06	55.42
dataX_v2	0.83	0.50	2.34	40.48	14.94	55.74
dataX	0.10	0.51	0.13	42.29	15.00	57.67
data-c-cc-nu-n_v2	0.05	0.87	0.05	119.27	33.48	153.28

train2	0.10	0.65	0.11	120.45	21.81	143.59
data-c-cc-nu-n	0.24	0.66	0.26	123.37	34.31	159.49
train1_v1	0.08	0.72	0.11	146.64	22.03	170.14
3d_dataset3	1.99	1.00	6.11	11923.00	399.94	12325.35
train3	0.14	0.91	0.23	318.46	51.66	377.31
data_circle	0.10	2.17	0.12	1765.74	606.09	2390.75
data_mix_uniform_normal	0.17	4.19	0.29	3250.87	295.99	3700.85
data_circle_2_10_2_12	0.92	6.63	2.75	3430.11	2273.90	5784.75
data_circle_5_10_8_12	2.71	10.23	5.71	2504.25	2403.26	5092.63
3d_dataset4	2.11	32.60	5.57	2714383.00	82117.80	2796865.30
data_circle1	0.29	16.69	0.47	3414.44	10983.40	15261.26
data_circle_1_20_1_15	0.77	19.08	3.68	8461.14	19980.85	30163.46

* Times for NOM include NC and OD times.

Table A8. Performance of clustering algorithms in terms of time (in seconds) for group 2 data sets

Data set	***k*-means**	**SL**	**DBSCAN**	**NC**	**OD**	**NOM***
D_0001	3.69	293.72	13.09	24832592.00	51118.54	24883768.88
D_0011	3.88	267.73	13.48	36493251.00	83338.22	36576925.47
D_0101	2.26	102.05	14.40	9769738.00	12538.38	9782294.18
D_0111	2.26	101.57	14.27	3263598.00	5548.29	3269162.48
D_0201	3.74	48.79	12.66	28430650.00	12642.69	28450350.29
D_0211	3.84	48.82	23.01	23344144.00	22100.90	23374030.94
D_1001	5.86	170.14	232.33	13781560.00	9587.78	13791169.13
D_1011	3.90	164.47	14.87	16609053.00	8218.61	16617291.82
D_1101	1.97	59.17	10.91	3026006.00	2351.49	3028447.80
D_1111	2.28	59.77	11.94	3960810.00	3641.22	3964510.91
D_1201	3.81	48.66	16.31	16772753.00	12758.56	16788467.91
D_1211	3.92	48.64	14.79	11819776.00	5800.89	11825587.75

* Times for NOM include NC and OD times.