Hybrid Algorithm for Noise-free High Density Clusters with Self-Detection of Best Number of Clusters

T. Karthikeyan[1], S. John Peter[2] and S.Chidambaranathan[3]

[1] *Department of Computer Science, PSG College of Arts and Science, Coimbatore, Tamil Nadu, India*
E-mail: t.karthikeyan.gasc@gmail.com

[2]*Department of Computer Science and Research Center*
St. Xavier's College, Palayamkottai, Tamil Nadu, India.
E-mail: jaypeeyes@rediffmail.com

[3]*Department of MCA*
St. Xavier's College, Palayamkottai, Tamil Nadu, India.
E-mail: scharan2009@rediffmail.com

Abstract

Clustering is a process of discovering group of objects such that the objects of the same group are similar, and objects belonging to different groups are dissimilar. A number of clustering algorithms exist that can solve the problem of clustering, but most of them are very sensitive to their input parameters. Minimum Spanning Tree clustering algorithm is capable of detecting clusters with irregular boundaries. A density-based notion of clusters which is designed to discover clusters of arbitrary shape. In this paper we propose a combined approach based on Minimum Spanning Tree based clustering and Density-based clustering for noise-free high density best number of clusters. The algorithm uses a new cluster validation criterion based on the geometric property of data partition of the data set in order to find the proper number of clusters at each level. The algorithm works in two phases. The first phase of the algorithm produces subtrees (noise-free clusters). The second phase finds high density clusters from the subtrees.

Keywords: *Center, Clustering, Cluster validity, Cluster Separation, Euclidean minimum spanning tree, Eccentricity, Outliers, Subtree,*

1. Introduction

One of the best known problems in the field of data mining is clustering. The problem of clustering is to partition a data set into groups (clusters) in such a way that the data elements within a cluster are more similar to each other than data elements in different clusters [18]. Clustering is the subject of active research in several fields such as statistics, pattern recognition, machine learning, and data mining. A wide variety of clustering algorithms have been proposed for different applications [2]. Much data of current interest to the scientific community can be modeled as graphs. A graph is sets of vertices, representing objects, connected together by edges, representing the relationship between objects. For example

World Wide Web can be modeled as a graph, where web pages are represented as vertices that are connected by an edge when one page contains a hyper link to another [1, 26].

Density-based clustering algorithms are designed to discover arbitrary-shaped clusters. Clusters are considered as dense regions of objects in the data space that are separated by regions of low density. DBSCAN [10] is a typical density-based algorithm. The core point in DBSCAN is that for a fixed radius, the neighborhood of each object in a cluster has to contain at least minimum number of other objects.

The density-based clustering algorithm can also be described as a search for the connected components in a graph where the vertices correspond to the core points in the data set, and there exits an (undirected) edge between two vertices (core points) if the distance between then is less than ε (ie each of them is in the neighborhood of the other point). The connected components of this graph correspond to the core points of each cluster.

Graph-theoretic approaches are able to discover non usual data structures. The simplest algorithm founded upon the Minimum Spanning Tree (MST) [42]. First Minimum Spanning Tree is built, where the nodes are objects and the edges sizes are the Euclidean distances in the objects space. The edges having the greatest size are removed to construct disjoint connected components.

An outlier is an observation of data that deviates from other observations so much that it arouses suspicious that was generated by a different mechanism from the most part of data [17]. Inlier, on the other hand, is defined as observation that is explained by underlying probability density function. In clustering, outliers are considered as noise observations that should be removed in order to make more reasonable clustering [18].

The outlier detection problem in some cases is similar to the classification problem. Clustering is a popular technique used to group similar data points or objects in groups or clusters [2]. Clustering is an important tool for outlier analysis. Several clustering–based outlier deduction techniques have been developed. Most of these techniques rely on the key assumption that normal objects belong to large and dense clusters, while outliers form very small clusters [30, 31]. The main concern of *clustering–based* outlier detection algorithms is to find clusters and outliers, which are often regarded as noise that should be removed in order to make more reliable clustering [20]. Some noisy points may be far away from the data points, whereas the others may be close. The far away noisy points would affect the result more significantly because they are more different from the data points. It is desirable to identify and remove the outliers, which are far away from all the other points in cluster [22]. So, to improve the clustering such algorithm uses the same process and functionality to solve both clustering and outlier discovery [11].

In this paper, we consider outliers as points, which are far from the most of other data. The proposed approach integrates minimum spanning tree based clustering and density-based approach for generating high density noise-free clusters. Here a vertex is defined as an outlier if it participates in at most *T* neighborhoods in **MST** graph, where threshold *T* is control parameter. We classify a vertex as outlier on basis of its *degree number* in the **MST** graph.

Clustering by minimal spanning tree can be viewed as a hierarchical clustering algorithm which follows a divisive approach. Using this method firstly **MST** is constructed for a given input. There are different methods to produce group of clusters. If the number of clusters *k* is given in advance, the simplest way to obtain *k* clusters is to sort the edges of minimum spanning tree in descending order of their weights and remove edges with first *k*-1 heaviest weights [4, 42].

Geometric notion of centrality are closely linked to facility location problem. The distance matrix *D* can computed rather efficiently using Dijkstra's algorithm with time complexity $O(|V|^2 \ln |V|)$ [38].

The *eccentricity* of a vertex *x* in *G* and radius ρ (G), respectively are defined as

$$e(x) = \max_{y \in V} d(x, y) \quad \text{and} \quad \rho(G) = \min_{x \in V} e(x)$$

The *center* of *G* is the set

$$\boldsymbol{C}(G) = \{x \in V \mid e(x) = \rho(G)\}$$

C (G) is the center to the "*emergency facility location problem*" which is always contain single block of *G*. The length of the longest path in the graph is called *diameter* of the graph *G*. we can define diameter D (G) as

$$D(G) = \max_{x \in V} e(x)$$

The *diameter* set of *G* is

$$Dia(G) = \{x \in V \mid e(x) = D(G)\}$$

All existing clustering Algorithm require a number of parameters as their inputs and these parameters can significantly affect the cluster quality. Our algorithm does not require a predefined cluster number. In this paper we want to avoid experimental methods and advocate the idea of need-specific as opposed to care-specific because users always know the needs of their applications. We believe it is a good idea to allow users to define their desired similarity within a cluster and allow them to have some flexibility to adjust the similarity if the adjustment is needed. Our Algorithm produces noise-free best number of clusters of *n*-dimensional points with a naturally approximate intra-cluster distance.

The problem of determining the correct number of clusters in a data set is perhaps the most difficult and ambiguous part of cluster analysis. The "true" number of clusters depends on the "level" on is viewing the data. Another problem is due to the methods that may yield the "correct" number of clusters for a "bad" classification [8]. Furthermore, it has been emphasized that mechanical methods for determining the optimal number of clusters should not ignore that the fact that the overall clustering process has an unsupervised nature and its fundamental objective is to uncover the unknown structure of a data set, not to impose one. For these reasons, one should be well aware about the explicit and implicit assumptions underlying the actual clustering procedure before the number of clusters can be reliably estimated or, otherwise the initial objective of the process may be lost. As a solution for this, Hardy [15] recommends that the determination of optimal number of clusters should be made by using several different clustering methods that together produce more information about the data. By forcing a structure to a data set, the important and surprising facts about the data will likely remain uncovered.

In some applications the number of clusters is not a problem, because it is predetermined by the context [16]. Then the goal is to obtain a mechanical partition for a particular data using a fixed number of clusters. Such a process is not intended for inspecting new and unexpected facts arising from the data. Hence, splitting up a homogeneous data set in a "fair" way is much more straightforward problem when compared to the analysis of hidden structures from heterogeneous data set. The clustering algorithms [23, 32] partitioning the data set in to *k* clusters without knowing the homogeneity of groups. Hence the principal goal of these clustering problems is not to uncover novel or interesting facts about data.

In this paper we propose a new method for clustering called **MSTDBCNFHDC** *(Minimum Spanning Tree based and Density-based Clustering for noise-free High density Clusters*). The purpose of our work is to find high-density clusters without any noise (unusual objects) by combining the Minimum Spanning Tree and Density-based clustering. To achieve this goal, we use the neighborhood of the vertices as clustering criteria instead of only their direct connections. Vertices are grouped into clusters by how they share neighbors.

The paper is organized as follows. We review the related works on Minimum Spanning Tree based clustering algorithms, density-based clustering algorithm and outlier detection in section 2. The notion of structure-connected clusters and **MSTDBCNFHDC** algorithm is described in section 3. Finally in conclusion we summarize the strength of our method and possible improvements.

2. Related work

MST based Clustering is the division of a graph into a set of sub-graphs, called clusters. More Specifically Given a graph $G = \{V, E\}$, where V is set of vertices and E is a set of edges between vertices, the goal of the graph partitioning is to divide G into k disjoint sub-graphs $G_i=\{V_i, E_i\}$ in which $V_i \cap V_j = \Phi$ for any, and $i \neq j$ and $V= \sum v_i$ where $i = 1...k$. The number of sub-graphs, k, may or may not be known a priori. In this paper, we focus on simple, undirected and weighted graphs.

The min-max cut method [9] seeks to partition a graph $G= \{V, E\}$ into two clusters A and B. The principle of min-max clustering is minimizing the number of connections between A and B and maximizing the number of connections within each. A cut is defined the number of edges that would have to be removed to isolate the vertices in cluster A from those in cluster B. The min-max cut algorithm searches for the clustering that creates two clusters whose cut is minimized and while maximizing the number of remaining edges.

Clustering by Minimal Spanning Tree can be viewed as a hierarchical clustering algorithm which follows the divisive approach. Clustering algorithm based on minimum and maximum spanning tree were extensively studied. Zhan [42] proposes to construct **MST** of point set and delete inconsistent edges – the edges, whose weights are significantly larger than the average weight of the nearby edges in the tree. Asano, Bhattacharya, Keil and Yao [3] gave optimal O *(n log n)* algorithm using maximum spanning trees for minimizing the maximum diameter of a bipartition. Asano, Bhattacharya, Keil and Yao also considered the clustering problem in which the goal to maximize the minimum inter-cluster distance. They gave a *k*-partition of point set removing the *k*-1 longest edges from the minimum spanning tree constructed from that point set [3]. The identification of inconsistent edges causes problem in the **MST** clustering algorithm. There exist numerous ways to divide clusters successively, but there is not suitable a suitable choice for all cases.

The **MST** clustering algorithm has been widely used in practice. Xu (Ying), Olman and Xu (Dong) [40] use MST as multidimensional gene expression data. They point out that **MST**-based clustering algorithm does not assume that data points are grouped around centers or separated by regular geometric curve. Thus the shape of the cluster boundary has little impact on the performance of the algorithm. They described three objective functions and the corresponding cluster algorithm for computing *k*-partition of spanning tree for predefined $k > 0$. The algorithm simply removes *k*-1 longest edges so that the weight of the subtrees is minimized. The second objective function is defined to minimize the total distance between the center and each data point in the cluster. The algorithm removes first *k*-1 edges from the tree, which creates a *k*-partitions.

Zahn [42] proposes to construct **MST** of point set and delete inconsistent edges – the edges, whose weights are significantly larger than the average weight of the nearby edges in the tree. Zahn's inconsistent measure is defined as follows. Let *e* denote an edge in the **MST** of the point set, v_1 and v_2 be the end nodes of *e, w* be the weight of *e*. A *depth neighborhood* ***N*** of an end node *v* of an edge *e* defined as a set of all edges that belong to all the path of length *d* originating from *v*, excluding the path that include the edge *e*. Let N_1 and N_2 be the depth *d* neighborhood of the node v_1 and v_2. Let $\hat{W}_{N1}$ be the average weight of edges in N_1 and σN_1 be its standard deviation. Similarly, let $\hat{W}_{N2}$ be the average weight of edges in N_2 and σN_2 be its standard deviation. The inconsistency measure requires one of the three conditions hold:

1. $w > \hat{W}N_1 + c \text{ x } \sigma N_1$ *or* $w > \hat{W}N_2 + c \text{ x } \sigma N_2$

2. $w > max(\hat{W}N_1 + c \text{ x } \sigma N_1 , \hat{W}N_2 + c \text{ x } \sigma N_2)$

3. $\dfrac{w}{max\,(c \text{ x } \sigma N_1 , c \text{ x } \sigma N_2)} > f$

where *c* and *f* are preset constants. All the edges of a tree that satisfy the inconsistency measure are considered inconsistent and are removed from the tree. This result in set of disjoint subtrees each represents a separate cluster. Paivinen [33] proposed a Scale Free Minimum Spanning Tree **(SFMST)** clustering algorithm which constructs scale free networks and outputs clusters containing highly connected vertices and those connected to them.

The selection of the correct number of clusters is actually a kind of validation problem. A large number of clusters provides a more complex "model" where as a small number may approximate data too much. Hence, several methods and indices have been developed for the problem of cluster validation and selection of the number of clusters [36, 25, 36, 37, 39]. Many of them based on the within and between-group distance.

Density-based algorithms try to separate the set *D* into subsets of similar densities. In the ideal case they can determine the cluster number *k* automatically and detect clusters of arbitrary shape and size. Representatives are **DBSCAN, MajorClust, or Chameleon.** The runtime of these algorithms is in magnitude of hierarchical algorithms, i, e., *O(n2),* or even *O(n log(n))* for low-dimensional data if efficient data structures are employed [5, 10, 14].

A density-based cluster algorithm operationalizes two mechanisms:

(1) one to define a region $R \subseteq D$, which forms the basis for density analyses;

(2) another to propagate density information (the provisional cluster label) of *R*.

In **DBSCAN** a region is defined as the set of points that lie in the *ε-neighborhood* of some point *p*. Cluster label propagation from *p* to the other points in *R* happens if *\R* exceeds a given *MinPts*-threshold.

In **MajorClust** a region is not of a fixed size but implicitly defined by the current clustering *C*. While **DBSCAN** uses the point concentration in *ε-neighborhoods* to estimate densities, **MajorClust** derives its density information from the attraction a cluster *C* exerts on some point *q,* which is computed as the sum of all similarity values $\vartheta(q,\ p),\ p \in C$. Cluster label propagation from *C* to *q* happens if the attraction of *C* with respect to *q* is maximum among the attraction values of all clusters in *C*. Observe that the "true" density values for some data set *D* evolve during the clustering process[5].

DBSCAN [10] operationalizes density propagation according to the principle "accessibility from a core point". Each points whose *ε-neighborhood* contains more points than *MinPts* is called a core point. Each point which lies in an ε- neighborhood of a core points p adopts the same cluster label as *p.* The DBSCAN-algorithm propagates this relation through the set *D.* The algorithm terminates if each points is either assigned to a certain cluster or classified as noise.

MajorClust operationalizes density propagation according to the principle .maximum attraction wins. The algorithm starts by assigning each point in *D* its own cluster. Within the following re-labeling steps, a point adopts the same cluster label as the majority of bits weighted neighbors. If several such clusters exist, one of them is chosen randomly. The algorithm terminates if no point changes its cluster membership.

There is no single universally applicable or generic outlier detection approach [30, 31]. Therefore there is many approaches have been proposed to deduct outliers. These approaches are classified into four major categories as *distribution-based, distance-based, density-based and clustering-based* [43].

Distribution-based approaches [19, 4] develop statistical models from the given data then apply a statistical test to determine if an object belongs to this model or not. Objects that have low probability to belong to the statistical model are declared as outliers. However, *distribution-based* approaches cannot be applied in multidimensional dataset because of the univariate in nature.

In the *distance-based* approach [27, 28, 29, 34], outliers are detected using a given distance measure on feature space, A point *q* in a data set is an outlier with respect to the parameters *M* and *d*, if there are less than *M* points within the distance *d* from *q,* where the values of *M* and *d* are determined by the user. The problem in distance–based approach is that it is difficult to determine the *M* and *d* values.

In *Density-based* methods [6] outlier is defined from local density of observation. These methods used different density estimation strategies. A low local density on the observation is an indication of a possible outlier. Brito et al [7] proposed a *Mutual k-Nearest-Neighbor* (**MkNN**) graph based approach. **MkNN** graph is a graph where an edge exits between vertices v_i and v_j if they both belong to each others k-neighborhood. **MkNN** graph is undirected and is special case of *k-Nearest-Neighbor* (**kNN**) graph, in which every node has pointers to its *k* nearest neighbors. Each connected component is considered as cluster, if it contains more than one vector and an outlier when connected component contains only one vector. Connected component with just one vertex is defined as an outlier.

Clustering-based approaches [30, 12, 20, 22], consider clusters of small sizes as outliers. In these approaches, small clusters (clusters containing significantly less points than other clusters) are considered as outliers. The advantage of *clustering- based* approaches is that they do not have to be supervised.

3. MSTDBCNFHDC Algorithm

Given a point set *S* in $\mathbf{E}^n$, the hierarchical method starts by constructing a Minimum Spanning Tree (**MST)** from the points in *S*. The weight of the edge in the tree is Euclidean distance between the two end points. So we named this **MST** as **EMST1.** Next the average weight $\hat{W}$ of the edges in the entire **EMST1** and its standard deviation σ are computed; any edge with $W > \hat{W} + \sigma$ or *current longest edge* is removed from the tree. This leads to a set of disjoint subtrees $S_T = \{T_1, T_2 \ldots\}$ *(divisive approach)*. Each of these subtrees T_i is treated as cluster. Oleksandr Grygorash et al proposed algorithm [32] which generates *k* clusters. The minimum spanning tree based algorithm [32] assumed the desired number of clusters in advance. In practice, determining the number of clusters is often coupled with discovering cluster structure. Hence we propose a new algorithm named; *Minimum Spanning Tree based and Density-based Clustering Algorithm for noise-free High-density Clusters (***MSTDBCNFHDC**), which does not require a predefined cluster number. The algorithm works in two phases. The first phase of the algorithm partitioned the **EMST1** into sub trees (clusters/regions). The centers of clusters or regions are identified using eccentricity of points. These points are a representative point for the each subtree S_T. A point c_i is assigned to a cluster *i* if $c_i \in T_i$. The group of center points is represented as $C = \{c_1, c_2 \ldots\ldots c_k\}$. These center points $c_1, c_2 \ldots . c_k$ are connected and again minimum spanning tree **EMST2** is constructed is shown in the Figure 4. This **EMST2** is used for finding best number of clusters. A Euclidean distance between pair of clusters can be represented by a corresponding weighted edge. Our algorithm is also based on the minimum spanning tree but not limited to two-dimensional points. There were two kinds of clustering problem; one that minimizes the maximum intra-cluster distance and the other maximizes the minimum inter-cluster distances. Our Algorithm produces best number of noise-free high-density clusters.

The structure of a vertex in **EMST** can be described by its neighborhood. Let $v \in V$, the structure of v is defined by its neighborhood denoted by Γ(v)

$$\Gamma(v) = \{w \in V \mid (v, w) \in E\} \cup \{v\} \qquad (1)$$

We normalize the number of common neighbors by computing distance of two neighbors using the function *dist (v, w)* the values of structural similarity will be computed.

When a member of a cluster shares a similar structure with one of its neighbors, their computed structural similarity will be large. We apply a threshold ε to the computed structural similarity when assigning cluster membership, formulized in the following ε-neighborhood definition.

$$N_\varepsilon(v) = \{w \in \Gamma(v) \mid dist(v, w) \geq \varepsilon\} \qquad (2)$$

When a vertex shares structural similarity with enough neighbors, it becomes a nucleus or seed for cluster. Such a vertex is called a core vertex. Core vertices are special classes of vertices that have a minimum of μ neighbors with a structural similarity that exceeds the threshold ε. From core vertices we grow the clusters. In This way the parameters μ and ε determine the clustering of graph. For a given ε. The minimal size of a cluster is determined by μ. Let $\varepsilon \in \Re$ and $\mu \in \aleph$, a vertex $v \in V$ is called a core w.r.t. ε and μ, if its ε-neighborhood contains at least μ vertices.

$$CORE_{\varepsilon,\mu}(v) \Leftrightarrow |N_\varepsilon(v)| \geq \mu \qquad (3)$$

We grow clusters from core vertices as follows. If a vertex is in ε-neighborhood of a core, it should be also in the same cluster. They share a similar structure and are connected. This idea is formulized in the following definition of direct structure reachability.

$$DirREACH_{\varepsilon,\mu}(v,w) \Leftrightarrow CORE_{\varepsilon,\mu}(v) \wedge w \in N_\varepsilon(v) \qquad (4)$$

Direct structure reachabiltiy is symmetric for any pair of cores. However, it is asymmetric if one of the vertices is not a core. The following definition is a canonical extension of direct structure reachability.

A vertex $w \in V$ is structure reachable from $v \in V$ w.r.t. ε and μ, if there is a chain of vertices $v_1, \ldots, v_n \in V, v_1 = v, v_n =$ w such that v_{i+1} is directly structure reachable from v_i, formally:

$$REACH_{\varepsilon,\mu}(v,w) \Leftrightarrow$$
$$\exists v_1, \ldots v_n \in V : v_1 = v \wedge v_n = w \wedge$$
$$\forall i \in \{1, \ldots n-1\} : DirREACH_{\varepsilon,\mu}(v_i, v_{i+1}) \qquad (5)$$

The structure reachability is transitive, but it is asymmetric. It is only symmetric for a pair of cores. More specifically, the structure-reachability is a transitive closure of direct structure reachability. A non-empty subset $C \subseteq V$ is called a structure-connected cluster w.r.t ε and μ, if all vertices in C are structure-connected and C is maximal w.r.t structure reachability. A clustering *P* of graph $G = <V, E>$ w.r.t. ε and μ consists of all structure connected clusters w.r.t. ε andμ in *G*.

To detect the outliers from the clusters we use the *degree number* of points in the clusters. For any undirected graph *G* the *degree* of a vertex *v*, written as *deg (v),* is equal to the number of edges in G which contains *v,* that is, which are incident on *v*[13].

We propose the following definition for outliers based on **MST,**

Definition 1: Given a **MST** for a data set *S*, outlier is a vertex *v*, whose *degree* is equal to 1, with *dist (v, Nearest-Neighbor (v)) > THR.*

where *THR* is a threshold value used as control parameter. The optimal number of subtrees (clusters) T_i, are created from the **EMST1** using the first phase of the **MSTDBCNFHDC** algorithm. Each T_i is treated as a **MST.** Then vertices *v,* which have *degree* 1 are identified. Then we find *Nearest-Neighbors* for the above vertices *v*. The *distance* between the vertices *v* and its nearest neighbor vertex is computed. If the computed *distance* exceeds the threshold value *THR* then the corresponding vertices are identified as an outlier is shown in the Fig 2.

When scanning the **EMST,** the edges are ordered from smaller to larger lengths. Then we define the threshold as:

$$THR = max\ (L_i - L_{i-1}) * t \qquad (6)$$

Where L_i is largest in the order and $t \in [0,1]$ is a user defined parameter.

Here, in this algorithm we use a cluster validation criterion based on the geometric characteristics of the clusters, in which only the inter-cluster metric is used. The **MSTDBCNFHDC** algorithm is a nearest centroid-based clustering algorithm, which creates region or subtrees (clusters/regions) of the data space. The algorithm partitions a set *S* of data of data *D* in data space in to *n* regions (clusters). Each region is represented by a centroid reference vector. If we let *p* be the centroid representing a region (cluster), all data within the region (cluster) are closer to the centroid *p* of the region than to any other centroid *q*:

$$R\ (p) = \{x \in D \mid dist(x, p) \le dist(x, q)\ \ \forall q\} \qquad (7)$$

Thus, the problem of finding the proper number of clusters of a dataset can be transformed into problem of finding the proper region (clusters) of the dataset. Here, we use the **MST** as a criterion to test the inter-cluster property. Based on this observation, we use a cluster validation criterion, called Cluster Separation (CS) in **MSTDBCNFHDC** algorithm [11].

Cluster separation (CS) is defined as the ratio between minimum and maximum edge of MST. ie

$$CS = E_{min} \,/\, E_{max} \qquad (8)$$

where E_{max} is the maximum length edge of **MST**, which represents two centroids that are at maximum separation, and E_{min} is the minimum length edge in the MST, which represents two centroids that are nearest to each other. Then, the CS represents the relative separation of centroids. The value of CS ranges from 0 to 1. A low value of CS means that the two centroids are too close to each other and the corresponding partition is not valid. A high CS value means the partitions of the data is even and valid. In practice, we predefine a threshold to test the CS. If the CS is greater than the threshold, the partition of the dataset is valid. Then again partitions the data set by creating subtree (cluster/region). This process continues until the CS is smaller than the threshold. At that point, the proper number of clusters will be the number of cluster minus one. The CS criterion finds the proper binary relationship among clusters in the data space. The value setting of the threshold for the CS will be practical and is dependent on the dataset. The higher the value of the threshold the smaller the number of clusters would be. Generally, the value of the threshold will be > 0.8[11]. Figure 3 shows the CS value versus the number of clusters in hierarchical clustering. The CS value < 0.8 when the number of clusters is 5. Thus, the proper number of clusters for the data set is 4.

Furthermore, the computational cost of CS is much lighter because the number of subclusters is small. This makes the CS criterion practical for the **MSTDBCNFHDC** algorithm when it is used for clustering large dataset.

In this paper, we focus on simple, undirected and weighted graph. Let $G = \{V, E\}$ be a graph, where V is a set of vertices; and E is a set of pairs of distinct vertices, called edges. $W(u, v)$ is the weight of the edge (u, v). The hierarchical method starts by constructing a Minimum Spanning Tree (**MST).** The weight of the edge in the tree is Euclidean distance between the two end points (vertices). So we named this **MST** as **EMST.** Xiaowei et al [41] proposed a method to find clusters, outliers and hubs based on undirected and un-weighted graph (Networks). Clustering algorithms using minimal spanning tree takes the advantage of **MST**. The **MST** ignores many possible connections between the data patterns, so the cost of clustering can be decreased. We modified the approach using Minimum Spanning Tree (**MST**) in order to find high density clusters. We propose a new definition for outliers based on the **EMST**.

```
Algorithm: MSTDBCNFHDC ( )
Input    : Point set S, $\mu$, $\varepsilon$ , THR
Output   : best number of high density noise-free clusters

Let $e$ be an edge in the EMST1 constructed from $S$
Let $W_e$ be the weight of $e$
Let $\sigma$ be the standard deviation of the edge weights
 in EMST1
Let $S_T$ be the set of disjoint subtrees of the EMST1
Let $n_c$ be the number of clusters

 1. Construct an EMST1 from $S$
 2. Compute the average weight of $\hat{W}$ of all the Edges from EMST1
 3. Compute standard deviation $\sigma$ of the edges
 4. $S_T = \emptyset$; $n_c = 1$
 5. Repeat
 6.   For each $e \in$ EMST1
 7.     If $(W_e > \hat{W} + \sigma)$ or (current longest edge $e$)
 8.       Remove $e$ from EMST1 which result $T'$, $a$ is new disjoint subtree
 9.       $S_T = S_T \cup \{T'\}$ // $T'$ is new disjoint subtree
10.       $n_c = n_c + 1$
11.       $C = \cup_{T_i} \in S_T \{C_i\}$
12.       Construct an EMST2 $T$ from $C$
13.       $E_{min}$ = get-min-edge ($T$)
14.       $E_{max}$ = get-max-edge ($T$)
15.       $CS = E_{min} / E_{max}$
16.     For p = $1$ to $|T_i|$ do
17.         If deg ($v_p$) == 1 and dist ($v_p$, Nearest-Neighbor ($v_p$)) > THR then remove $v_p$ from $T_i$
18. For each unclassified vertex $v \in V$ in $T_i$ do
19.   If $CORE_{\varepsilon,\mu}(v)$ then
20.        Generate new clusterID
21.        Insert all $x \in N_\varepsilon(v)$ in to queue $Q$
```

```
22.         While $Q \neq 0$ do
23.            $y$ = remove($Q$)
24.            $R = \{x \in V \mid DirREACH_{\varepsilon,\mu}(y,x)\}$
25.                For $x \in R$ do
26.                   If $x$ is unclassified then assign current clusterID to $x$
27.                      Insert $x$ into queue $Q$.
28. Until $CS < 0.8$
29. Return best number of noise-free high-density clusters
```

Here we describe the **MSTDBCNFHDC** algorithm, which implements the search for noise-free high density clusters, outliers. The **MSTDBCNFHDC** algorithm first identifies outliers. Then it visiting each vertex once to find structure-connected high density clusters.

Figure 1 shows a typical example of **EMST1** constructed from point set S, in which inconsistent edges are removed to create subtree (clusters/regions). Our algorithm finds the center of the each cluster, which will be useful in many applications. Figure 2 shows the possible distribution of the points in the two cluster structures with their center vertex as 5 and 3, outlier vertex as 2 and 7.

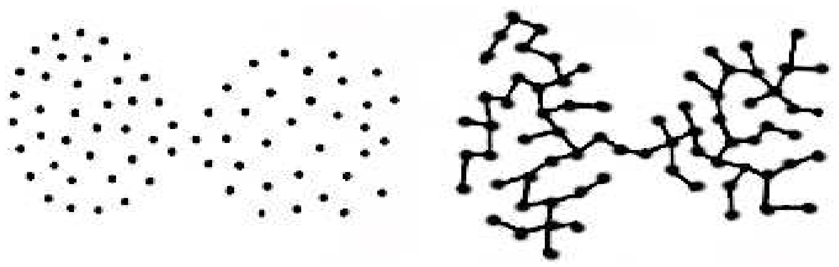
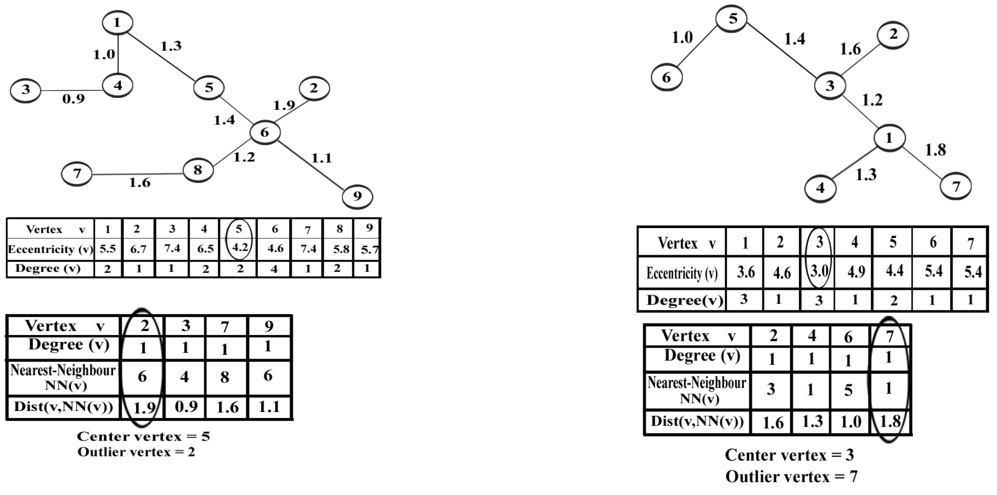


Figure 1. EMST1 - Clusters connected through a point



Vertex v	1	2	3	4	5	6	7	8	9
Eccentricity (v)	5.5	6.7	7.4	6.5	4.2	4.6	7.4	5.8	5.7
Degree (v)	2	1	1	2	2	4	1	2	1

Vertex v	2	3	7	9
Degree (v)	1	1	1	1
Nearest-Neighbour NN(v)	6	4	8	6
Dist(v,NN(v))	1.9	0.9	1.6	1.1

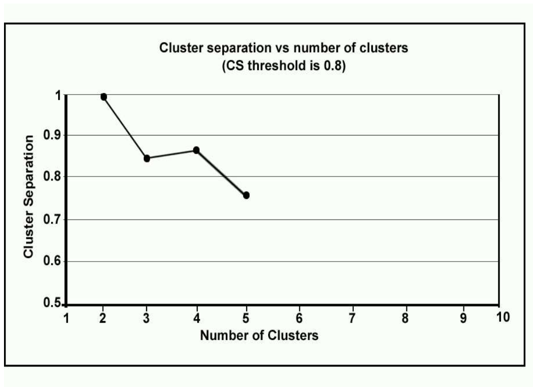
Center vertex = 5
Outlier vertex = 2

Vertex v	1	2	3	4	5	6	7
Eccentricity (v)	3.6	4.6	3.0	4.9	4.4	5.4	5.4
Degree(v)	3	1	3	1	2	1	1

Vertex v	2	4	6	7
Degree (v)	1	1	1	1
Nearest-Neighbour NN(v)	3	1	5	1
Dist(v,NN(v))	1.6	1.3	1.0	1.8

Center vertex = 3
Outlier vertex = 7

Figure 2. Two Clusters/regions (MST) with center points 5 and 3 (outliers 2 and 7)



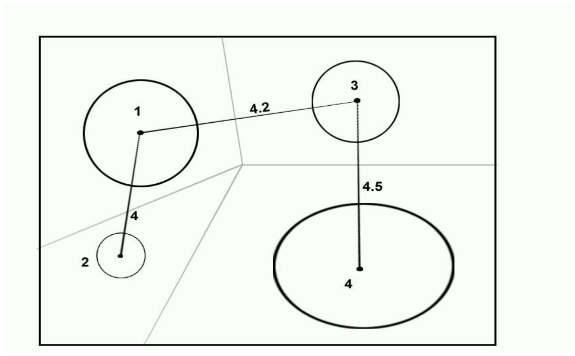


Figure 3. Number of Clusters vs. Cluster Separation **Figure 4. EMST2 From 4 region/cluster center points**

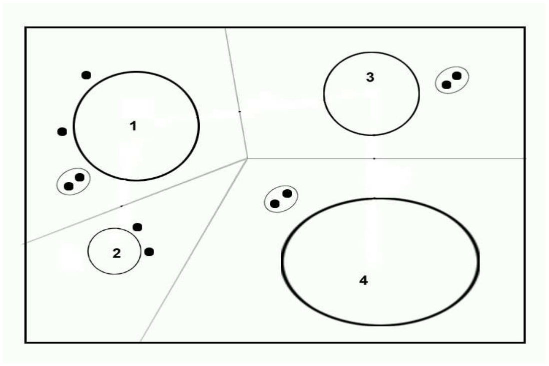


Figure 5. Four Clusters with outliers as black spots

Our **MSTDBCNFHDC** algorithm works in two phases. The first phase of the algorithm (lines 1-17) uses *divisive approach* of hierarchical clustering. Euclidean minimum spanning tree **EMST1** is constructed in line 1. The average and standard deviation of the weighted edges of the Euclidean minimum spanning tree are computed to find inconsistent edges are specified in the lines 2-3. The inconsistent edges are identified and removed from Euclidean minimum spanning tree **EMST1** in order to generate subtree *T'* is specified in the lines 7-9. The center of each subtree is computed. Lines 13-15 in the algorithm are used find the value of cluster separation (CS). This value is useful to find best number of clusters. Outliers are identified and removed from clusters lines (16-17).

The second phase of the algorithm is used for finding high density clusters (lines 19-27). The output of the first phase (subtree/cluster) is given as input to the second phase. The subtree is again an **EMST.** Then it performs one pass of an **EMST** and finds all structure-connected clusters for a given parameters settings. At the beginning all the vertices are labeled as unclassified. The second phase of the **MSTDBCNFHDC** algorithm classifies each vertex either a member of a cluster or non-member. For each vertex that is not yet classified, **MSTDBCNFHDC** checks whether this vertex is a core (line 19). If the vertex is a core, a new cluster is expanded from this vertex (lines 20-27). To find a new cluster **MSTDBCNFHDC** starts with an arbitrary core *v* and search for all vertices that are structure-reachable from *v* (line 24). New Cluster ID is generated which will be assigned to all vertices found in (line 26). **MSTDBCNFHDC** begins by inserting all vertices in ε-neighborhood of vertex *v* in to a queue. For each vertex in a queue it computes all directly reachable vertices and inserts those vertices into queue which are still unclassified. This is repeated until the queue is empty. The second phase of the algorithm finds best number of high density clusters.

4. Conclusion

Our **MSTDBCNFHDC** clustering algorithm does not assumes any predefined cluster number. The algorithm gradually finds clusters with center for each cluster. These clusters ensure guaranteed intra-cluster similarity. Our algorithm does not require the users to select and try various parameters combinations in order to get the desired output. Our **MSTDBCNFHDC** clustering algorithm detects outliers from cluster ant it uses a new cluster validation criterion based on the geometric property of partitioned regions/clusters to produce best number of "true" clusters with center for each of them. The inter-cluster distances between centers of clusters/regions are used to find best number of noise-free clusters. All of these look nice from theoretical point of view. However from practical point of view, there is still some room for improvement for running time of the clustering algorithm. This could perhaps be accomplished by using some appropriate data structure. In the future we will explore and test our proposed clustering algorithm in various domains. We will further study the rich properties of **EMST**-based clustering methods in solving different clustering problems. In Future we will try to apply our algorithm to various domains including bioinformatics, medical image processing and geographical information system.

References

[1] R. Albert, H. Jeong, and A.-L. Barabási, "Diameter of the world-wide web." *Nature* 401, 130–131 1999.

[2] Anil K. Jain, Richard C. Dubes "Algorithm
for Clustering Data", *Michigan State University, Prentice Hall, Englewood Cliffs,New Jersey* 07632.1988.

[3] T. Asano, B. Bhattacharya, M.Keil and F.Yao. "Clustering Algorithms based on minimum and maximum spanning trees". *In Proceedings of the 4th Annual Symposium on Computational Geometry*, Pages252-257,1988.

[4] V.Barnett and T.Lewis, "Outliers in Statistical Data", *John Wiley,* 1994.

[5] Benno Stein and Oliver Niggemann. "On the Nature of Structure and its Identification". In Peter Widmayer, Gabriele Neyer, and Stefan Eidenbenz, editors, *Graph-Theoretic Concepts in Computer Science*, volume 1665 LNCS of *Lecture Notes in Computer Science*, pages 122.134. Springer, June 1999. ISBN 3-540-66731-8.

[6] M. Breunig, H.Kriegel, R.Ng and J.Sander, Lof: "Identifying density-based local outliers". *In Proceedings of 2000 ACM SIGMOD International Conference on Management of Data. ACM Press,* pp 93-104, 2000.

[7] M. R. Brito, E. L. Chavez, A. J. Quiroz, and J. E. Yukich. "Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection". *Statistics & Probability Letters,* 35(1):33-42, 1997.

[8] B.Custem and I.Gath,,"Detection of Outliers and Robust Estimation using Fuzzy clustering", *Computational Statistics & data Analysis* 15,pp.47-61, 1993.

[9] C. Ding, X. He, H. Zha, M. Gu, and H. Simon, "A min-max cut algorithm for graph partitioning and data clustering", *Proc. of ICDM* 2001.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise". In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD96)*, 1996.

[11] Feng Luo,Latifur Kahn, Farokh Bastani, I-Ling Yen, and Jizhong Zhou, "A dynamically growing self-organizing tree(DGOST) for hierarchical gene expression profile",Bioinformatics,Vol 20,no 16, pp 2605-2617, 2004.

[12] Gath and A.Geva, "Fuzzy Clustering for the estimation of the Parameters of the components of Mixtures of Normal distribution", *Pattern Recognition letters*, 9, pp.77-86, 1989.

[13] Gary Chartrand and Ping Zhang "Introduction to Graph Theory", *Tata MgrawwHill, Paperback*-2008.

[14] G. Hamerly and C. Elkan, "Learning the k in k-means, in Advances in Neural Information Processing Systems" 16, S. Thrun, L. Saul, and B. Schölkopf, eds., *MIT Press, Cambridge, MA,* 2004.

[15] A. Hardy, "On the number of clusters", *Computational Statistics and Data Analysis,* 23, pp. 83–96, 1996.

[16] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: Data mining, inference and prediction", *Springer-Verlag, 2001.*

[17] B. Ghosh-Dastidar and J.L. Schafer, "Outlier Detection and Editing Procedures for Continuous Multivariate Data". *ORP Working Papers,* September 2003.

[18] S. Guha, R. Rastogi, and K. Shim. "CURE an efficient clustering algorithm for large databases". *In Proceeding of the 1998 ACM SIGMOD Int. Conf. on Management of Data , pp 73-84, Seattle, Washington,* 1998.

[19] D.Hawkins, "Identifications of Outliers", *Chapman and Hall,* London, ,1980.

[20] Z. He, X. Xu and S. Deng, "Discovering cluster-based Local Outliers", *Pattern Recognition Letters,* Volume 24, Issue 9-10, pp 1641 – 1650, June 2003.

[21] H.Gabow, T.Spencer and R.Rarjan. "Efficient algorithms for finding minimum spanning trees in undirected and directed graphs", *Combinatorica,* 6(2):pp 109-122, 1986.

[22] M. Jaing, S. Tseng and C. Su, "Two-phase Clustering Process for Outlier Detection", *Pattern Recognition Letters,* Volume 22, Issue 6 – 7, pp 691 – 700, May 2001.

[23] S. John Peter, S.P. Victor, "A Novel Algorithm for Dual similarity clusters using Minimum spanning tree". *Journal of Theoretical and Applied Information technology,* Vol.14. No.1 pp 60-66, 2010.

[24] D. Karger, P. Klein and R. Tarjan, "A randomized linear-time algorithm to find minimum spanning trees", *Journal of the ACM,* 42(2):321-328, 1995.

[25] G. Karypis, E.-H. Han, and V. Kumar. "Chameleon: A hierarchical clustering algorithm using dynamic modeling", *Technical Report* Paper No. 432, University of Minnesota, Minneapolis, 1999.

[26] J. M. Kleinberg, S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "The Web as a graph: Measurements,models and methods." *In Proceedings of the International Conference on Combinatorics and Computing, number 1627 in Lecture Notes in Computer Science,* pp. 1–18, *Springer, Berlin* 1999.

[27] E. Knorr and R. Ng, "A Unified Notion of Outliers: Properties and Computation". *In Proceedings of the Third International Conference on Knowledge Discovery and Data Mining,* pages 219 – 222, August 1997.

[28] E..Knorr and R.Ng, "Algorithms for Mining Distance-based Outliers in Large Data sets", *Proc.the 24th International Conference on Very Large Databases(VLDB)*,pp.392-403, 1998.

[29] E.Knorr, R.Ng and V.Tucakov, "Distance- Based Outliers: Algorithms and Applications", *VLDB Journal,* 8(3-4):237-253, 2000.

[30] A.Loureiro, L.Torgo and C.Soares, "Outlier detection using Clustering methods: A data cleaning Application", *in Proceedings of KDNet Symposium on Knowledge-based systems for the Public Sector.* Bonn, Germany, 2004.

[31] K.Niu, C.Huang, S.Zhang and J.Chen, "ODDC: Outlier Detection Using Distance Distribution Clustering", *T.Washio et al. (Eds.): PAKDD 2007 Workshops, Lecture Notes in Artificial Intelligence (LNAI)* 4819,pp.332-343,*Springer-Verlag,* 2007.

[32] Oleksandr Grygorash, Yan Zhou, Zach Jorgensen. "Minimum spanning Tree Based Clustering Algorithms". *Proceedings of the 18th IEEE International conference on tools with Artificial Intelligence (ICTAI'06)* 2006.

[33] N. Paivinen, "Clustering with a minimum spanning of scale-free-like structure".*Pattern Recogn. Lett.,*26(7): 921-930, 2005.

[34] S. Ramaswamy, R. Rastogi and K. Shim, "Efficient Algorithms for Mining Outliers from Large Data Sets". *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data,* Volume 29, Issue 2, pages 427 – 438, May2000.

[35] D. M. Rocke and J. J. Dai, "Sampling and subsampling for cluster analysis in data mining: With applications to sky survey data", *Data Mining and Knowledge Discovery,* 7, pp. 215–232, 2003.

[36] S. Salvador and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms", *in Proceedings Sixteenth IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2004, Los Alamitos, CA, USA*, *IEEE Computer Society,* pp. 576–584 , 2004.

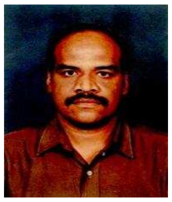
[37] S. Still and W. Bialek, "How many clusters?*" , An information-theoretic perspective*, *Neural Computation*, 16, pp. 2483–2506, 2004.

[38] Stefan Wuchty and Peter F. Stadler. "Centers of Complex Networks". 2006

[39] C. Sugar and G. James, “Finding the number of clusters in a data set *”, An information theoretic approach*, Journal of the American Statistical Association, 98 pp. 750–763, 2003.
[40] Y.Xu, V.Olman and D.Xu. “Minimum spanning trees for gene expression data clustering”, *Genome Informatics*,12:24-33, 2001.

[41] Xiaowei Xu, Nurcan Yuruk Zhidan Feng and Thomas A.J. Schweiger, “ SCAN: A Structural Clustering Algorithm for Networks”, *SIGKDD,* San Jose, CA, US, 2007.
[42] C. Zahn. “Graph-theoretical methods for detecting and describing gestalt clusters”, *IEEE Transactions on Computers*, C-20:68-86, 1971
[43] J. Zhang and N. Wang, “Detecting outlying subspaces for high-dimensional data: the new task, Algorithms and Performance”, *Knowledge and Information Systems,* 10(3):333-555, 2006.

Authors



Thirunavu Karthikeyan received his graduate degree in Mathematics from Madras University in 1982. Post graduate degree in Applied Mathematics from Bharathidasan University in 1984. Received Ph.D., in Computer Science from Bharathiar University in 2009. Presently he is working as a Associate Professor in Computer Science Department of P.S.G. College of Arts and Science, Coimbatore. His research interests are Image Coding, Medical Image Processing and Data mining. He has published many papers in national and international conferences and journals. He has completed many funded projects with excellent comments. He has contributed as a program committee member for a number of international conferences. He is the review board member of various reputed journals. He is a board of studies member for various autonomous institutions and universities. He can be contacted at t.karthikeyan.gasc@gmail.com

S. John Peter is working as Assistant professor in Computer Science, St.Xavier’s college (Autonomous), Palayamkottai, Tirunelveli. He earned his M.Sc degree from Bharadhidasan University, Trichirappalli. He also earned his M.Phil from Bharadhidasan University, Trichirappalli. Now he is doing Ph.D in Computer Science at Manonmaniam Sundaranar University, Tirunelveli. He has published research papers on clustering algorithm in various international journals. E-mail: jaypeeyes@rediffmail.com

S.Chidambaranathan, received his post graduate degree in Mathematics from Madurai Kamaraj University, Madurai. He also earned post graduate degree in Computer Application and M.Phil in computer Science from Manonmaniam Sundaranar University, Tirunelveli. Presently he is working as Assistant Professor in the Department of MCA, St. Xavier’s College (Autonomous), Palayamkottai, Tamil Nadu. He is an author for many books including “PHP for beginners”, “Learning XML” and “Everything HTML”. He has published many research papers in National, International journals and conference proceedings. He can be contacted at E-mail: scharan2009@rediffmail.com