Meta Similarity Noise-free Clusters Using Dynamic Minimum Spanning Tree with Self-Detection of Best Number of Clusters

T. Karthikeyan[1], S. John Peter[2]
[1] Department of Computer Science, PSG College of Arts and Science,
Coimbatore, Tamil Nadu, India
[2]Department of Computer Science and Research Center, St. Xavier's College,
Palayamkottai, Tamil Nadu, India.
[1] t.karthikeyan.gasc@gmail.com, [2]jaypeeyes@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. Detecting outlier in database (as unusual objects) is a big desire. In data mining detection of anomalous pattern in data is more interesting than detecting inliers. In this paper we propose a Minimum Spanning Tree based clustering algorithm for noise-free or pure clusters. The algorithm constructs hierarchy from top to bottom. At each hierarchical level, it optimizes the number of cluster, from which the proper hierarchical structure of underlying data set can be found. 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 create clusters with guaranteed intra-cluster similarity, where as the second phase of the algorithm create dendrogram using the clusters as objects with guaranteed inter-cluster similarity. The first phase of the algorithm uses divisive approach, where as the second phase uses agglomerative approach. In this paper we used both the approaches in the algorithm to find Best number of Meta similarity clusters.

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

1. INTRODUCTION

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 [1]. Inliers, 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 [2]. Outlier may be erroneous or real in the following sense. Real outliers are observations whose actual values are very different than those observed for rest of the data and violate plausible relationship among variables. Outliers can often be individual or groups of clients exhibiting behavior outside the range of what is considered normal. Outliers can be removed or considered separately in *regression modeling* to improve accuracy which can be considered as benefit of outliers. Identifying them prior to modeling and analysis is important [3]. In clustering–based methods, outlier is defined as observation that does not fit to the overall clustering pattern [4].

The importance of outlier detection is due to the fact that outliers in the data translate to significant (and often critical) information in a wide variety of application domains. For example, an anomalous traffic pattern in a computer network could mean that a hacked computer is sending out sensitive data to an unauthorized destination. In public health data, outlier detection techniques are widely used to detect anomalous pattern in patient medical records which could be symptoms of new diseases. Similarly, outliers in credit card transaction data could indicate credit card theft or misuse. Outliers can also translate to critical entities such as in military surveillance, whereas the presence of unusual region in a satellite image of enemy are could indicate enemy troop movement. Or anomalous readings from space craft would signify a fault in some of the craft. Outlier detection has been found to be directly applicable in large number of domains.

Many data-mining algorithms find outliers as a side-product of clustering algorithms. However these techniques define outlier as points, which do not lie in clusters. Thus, the techniques implicitly define outliers as the background noise in which the clusters are embedded. Another class of techniques defines outlier as points, which are neither a part of a cluster nor part of background noise; rather they are specifically points which behave very differently from the norm [5].

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 [6]. 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 [7, 8]. 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 [9]. 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 [10]. So, to improve the clustering such algorithm use the same process and functionality to solve both clustering and outlier discovery [11].

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 [12]. Furthermore, it has been emphasized that mechanical methods for determining the best number of 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 [12] recommends that the determination of best 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 [13]. 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 [14, 15] 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.

Given a connected, undirected graph $G = (V, E)$, where V is the set of nodes, E is the set of edges between pairs of nodes, and a weight $w(u, v)$ specifying weight of the edge (u, v) for each edge $(u, v) \in E$. A spanning tree is an acyclic subgraph of a graph G, which contains all vertices from G. The Minimum Spanning Tree (**MST**) of a weighted graph is minimum weight spanning tree of that graph. Several well established **MST** algorithms exist to solve minimum spanning tree problem [16, 17, 18]. The cost of constructing a minimum spanning tree is $O(m \log n)$, where m is the number of edges in the graph and n is the number of vertices. More efficient algorithm for constructing **MST**s have also been extensively researched [19, 20, 21]. These algorithms promise close to linear time complexity under different assumptions. A Euclidean minimum spanning tree (**EMST**) is a spanning tree of a set of n points in a metric space ($\mathbf{E}^n$), where the length of an edge is the Euclidean distance between a pair of points in the point set.

The hierarchical clustering approaches are related to graph theoretic clustering. 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. The **MST** based clustering algorithm is known to be capable of detecting clusters with various shapes and size [22]. Unlike traditional clustering algorithms, the **MST** clustering algorithm does not assume a spherical shapes structure of the underlying data. The **EMST** clustering algorithm [23, 16] uses the Euclidean minimum spanning tree of a graph to produce the structure of point clusters in the n-dimensional Euclidean space. Clusters are detected to achieve some measures of best number of identity, such as minimum intra-cluster distance or maximum inter-cluster distance [24]. The **EMST** algorithm has been widely used in practice.

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 [24, 25].

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 case-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 clusters of n-dimensional points with a naturally approximate intra-cluster distance.

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 $O(|V|^2 \ln |V|)$ [26].
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

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

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

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

An important objective of hierarchical cluster analysis is to provide picture of data that can easily be interpreted. A picture of a hierarchical clustering is much easier for a human being to comprehend than is a list of abstract symbols. A *dendrogram* is a special type of tree structure that provides a convenient way to represent hierarchical clustering. A dendrogram consists of layers of nodes, each representing a cluster.

Hierarchical clustering is a sequence of partitions in which each partition is nested into the next in sequence. An Agglomerative algorithm for hierarchical clustering starts with disjoint clustering, which places each of the *n* objects in an individual cluster [6]. The hierarchical clustering algorithm being employed dictates how the proximity matrix or proximity graph should be interpreted to merge two or more of these trivial clusters, thus nesting the trivial clusters into second partition. The process is repeated to form a sequence of nested clustering in which the number of clusters decreases as a sequence progress until single cluster containing all *n* objects, called the *conjoint clustering*, remains[4].

Nearly all hierarchical clustering techniques that include the tree structure have two short comings: (1) they do not properly represent hierarchical relationship and (2) once the data are assigned improperly to a given cluster it cannot later reevaluate and placed in another cluster.

In this paper, we propose a new algorithm *Dynamically Growing Euclidean Minimum Spanning Tree Algorithm for Noise-free Meta Clusters* (**DGEMSTNFMC**) which can overcome these shortcomings. The **DGEMSTNFMC** algorithm optimizes the number of clusters at each hierarchical level with the cluster validation criteria during the minimum spanning tree construction process. Then the hierarchy constructed by the algorithm can properly represent the hierarchical structure of the underlying dataset, which improves the accuracy of the final clustering result.

Our **DGEMSTNFMC** clustering algorithm addresses the issues of undesired clustering structure and unnecessary large number of clusters. Our algorithm does not require a predefined cluster number. The algorithm constructs an **EMST** of a point set and removes the inconsistent edges that satisfy the inconsistence measure. The process is repeated to create a hierarchy of clusters until best number of numbers of noise-free clusters (regions) is obtained. Hence the title! In section 2 we review some of the existing works on outliers and graph based clustering algorithms. In Section 3 we propose **DGEMSTNFMC** algorithm which produces best number of noise-free clusters with dendrogram. Hence we named this new cluster as *Best number of Meta similarity noise-free clusters*. Finally in conclusion we summarize the strength of our method and possible improvements.

2. RELATED WORK

There is no single universally applicable or generic outlier detection approach [7, 8]. 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* [27]. Here we discuss about density-based and clustering-based outlier detection approaches.

In *Density-based* methods 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 [28] 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. The problem with this approach is that outlier too close to inliers, can be misclassified. *Density-based* approaches [29, 30] compute the density of the regions in the data and declare the objects in low dense regions as outliers.

Clustering-based approaches [7, 31, 32, 10] 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.

Jiang et al., [10] proposed a two-phase method to detect outliers. In the first phase, clusters are produced using modified K-means algorithm, and then in the second phase, an Outlier-Finding Process **(OFP)** is proposed. The *small clusters* are selected and regarded as outliers. *Small cluster* is defined as a cluster with fewer points than half the average number of points in the *k* number of clusters. Loureio [7] proposed a method for detecting outlier. Hierarchical clustering technique is used for detecting outliers. The key idea is to use the size of the resulting clusters as indicators of the presence of outliers. Almedia [33] is also used similar approach for detecting outliers. Using the K-means clustering algorithm Yoon [34] proposed a method to detect outliers.

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. Avis [35] found an $O(n^2 \log^2 n)$ algorithm for the min-max diameter-2 clustering problem. Asano, Bhattacharya, Keil and Yao [24] later gave best number of O $(n \log n)$ algorithm using maximum spanning trees for minimizing the maximum diameter of a bipartition. The problem becomes NP-complete when the number of partitions is beyond two [36]. 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 [24]. 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.

Zahn [22] 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 *are* 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 is its standard deviation. The inconsistency measure requires one of the three conditions hold:

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

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

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

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 [37] 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 **MST** clustering algorithm has been widely used in practice. Xu (Ying), Olman and Xu (Dong) [25] 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.

The clustering algorithm proposed by S.C. Johnson [38] uses proximity matrix as input data. The algorithm is an agglomerative scheme that erases rows and columns in the proximity matrix as old clusters are merged into new ones. The algorithm is simplified by assuming no ties in the proximity matrix. Graph based algorithm was proposed by Hubert [39] using single link and complete link methods. He used threshold graph for formation of hierarchical clustering. An algorithm for single-link hierarchical clustering begins with the minimum spanning tree (MST) for G (∞), which is a proximity graph containing *n(n-1)/2* edge was proposed by Gower and Ross [40]. Later Hansen and DeLattre [9] proposed another hierarchical algorithm from graph coloring.

Many different methods for determining the number of clusters have been developed. Hierarchical clustering methods provide direct information about the number of clusters by clustering objects on a number of different hierarchical levels, which are then presented by a graphical tree structure known as *dendrogram.* One may apply some external criteria to validate the solutions on different levels or use the *dendrogram* visualization for determining the best cluster structure.

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 [41, 42, 43, 44, 45]. Many of them based on the within-and between-group distance.

3. OUR CLUSTERING ALGORITHM

A tree is a simple structure for representing binary relationship, and any connected components of tree is called *subtree*. Through this **MST** representation, we can convert a multi-dimensional clustering problem to a tree partitioning problem, ie., finding particular set of tree edges and then cutting them. Representing a set of multi-dimensional data points as simple tree structure will clearly lose some of the inter data relationship. However many clustering algorithm proved that no essential information is lost for the purpose of clustering. This is achieved through rigorous proof that each cluster corresponds to one subtree, which does not overlap the representing subtree of any other cluster. Clustering problem is equivalent to a problem of identifying these subtrees through solving a tree partitioning problem. The inherent cluster structure of a point set in a metric space is closely related to how objects or concepts are embedded in the point set. In practice, the approximate number of embedded objects can sometimes be acquired with the help of domain experts. Other times this information is hidden and unavailable to the clustering algorithm. In this section we present clustering algorithm which produce best number of clusters.

3.1 DGEMSTNFMC Clustering 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 minimum spanning tree based clustering algorithm [15] which generates *k* clusters. Our previous algorithm [14] generates *k* clusters with centers, which used to produce Meta similarity clusters. Both of the minimum spanning tree based algorithms 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, *Dynamically Growing Minimum Spanning Tree for Noise-free Meta Clustering* algorithm *(***DGEMSTNFMC**), 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 ....c_k are connected and again minimum spanning tree **EMST2** is constructed is shown in the Figure 4. 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 clusters with both intra-cluster and inter-cluster similarity. The Second phase of the algorithm converts the minimum spanning tree **EMST2** into *dendrogram*, which can be used to interpret about inter-cluster distances. This new algorithm is neither single link clustering algorithm (SLCA) nor complete link clustering algorithm (CLCA) type of hierarchical clustering, but it is based on the distance between centers of clusters. This approach leads to new developments in hierarchical clustering. The level function, *L,* records the proximity at which each clustering is formed. The levels in the *dendrogram* tell us the least amount of similarity that points between clusters differ. This piece of information can be very useful in several medical and image processing applications.

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*[46].

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.

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

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

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

The best number of subtrees (clusters) T_i, are created from the **EMST1** using the first phase of the **DGEMSTNFMC** 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.

Here, we use a cluster validation criterion based on the geometric characteristics of the clusters, in which only the inter-cluster metric is used. The **DGEMSTNFMC** 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*:

$$R(p) = \{x \in D \mid dist(x, p) \le dist(x, q) \ \forall q\} \qquad (2)$$

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 **DGEMSTNFMC** algorithm [47].

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

$$CS = E_{min} / E_{max} \qquad (3)$$

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 data set 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[47]. If the CS value is less than 0.8, then the created clusters are not well separated and we may not get the optimal number of clusters. 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. Further more, the computational cost of CS is much lighter because the number of subclusters is small. This makes the CS criterion practical for the **DGEMSTNFMC** algorithm when it is used for clustering large dataset.

Algorithm: DGEMSTNFMC ( )

```
Input   : S the point set
Output  : dendrogram with best number of Noise-free clusters

Let e1 be an edge in the EMST1 constructed from S
Let e2 be an edge in the EMST2 constructed from C
Let $W_e$ be the weight of e1
Let σ be the standard deviation of the edge weights in EMST1
Let $S_T$ be the set of disjoint subtrees of 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 σ of the edges
    from EMST1
4.  $S_T = ø$; $n_c = 1$; $C = ø$;
5.  Repeat
6.    For each $e1 \in$ EMST1
7.      If ($W_e > \hat{W} + σ$) or (current longest edge e1)
8.         Remove e1 from EMST1
9.         $S_T = S_T \cup \{ T' \}$ // T' is new disjoint
           Subtree (regions)
10.        $n_c = n_c + 1$
11.        Compute the center $C_i$ of $T_i$ using
           eccentricity of points and THR value
12.        $C = \cup_{T_i \in S_T} \{C_i\}$
13.        Construct an EMST2 T from C
14.        $E_{min}$ = get-min-edge (T)
15.        $E_{max}$ = get-max-edge (T)
16.        $CS = E_{min} / E_{max}$
17.        For p = 1 to $|T_i|$ do
18.        If deg ($v_p$) == 1 and dist ($v_p$, Nearest-
           Neighbor ($v_p$)) > THR then remove $v_p$
           from $T_i$
19. Until CS < 0.8
20. Begin with disjoint clusters with level L (0) =
     0 and sequence number m = 0
21. While (T has some edge)
22.     e2 = get-min-edge(T) // for least dissimilar
        pair of clusters
23.     (a, b) = get-vertices (e2)
24.     Increment the sequence number m = m + 1,
         merge the clusters (a) and (b), into single
        cluster to form next clustering m and set
        the level of this cluster to L(m) = e2;
25.     Update T by forming new vertex by
        combining the vertices a, b;
26. Return dendrogram for best number of
     Noise-free 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. Our algorithm will find best number of clusters or cluster structures. Figure 2 shows the possible distribution of the points in the two cluster structures with their center vertex as 5 and 3.

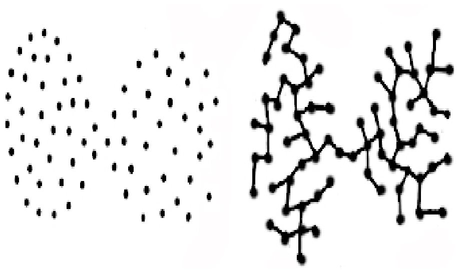
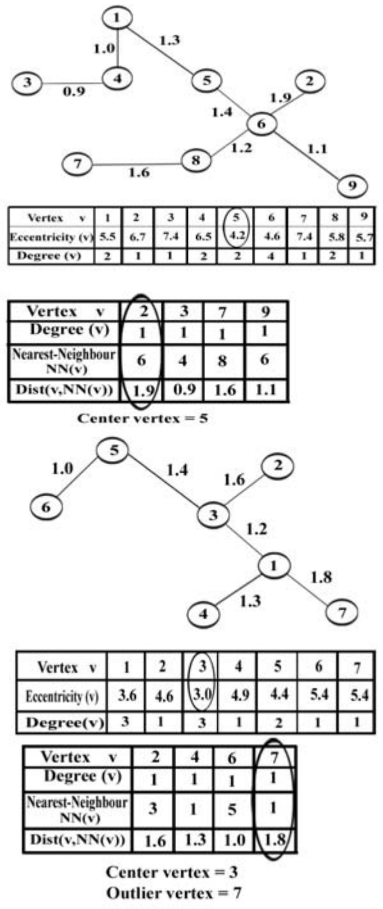


Figure 1: Clusters connected through points -EMST1



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

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)

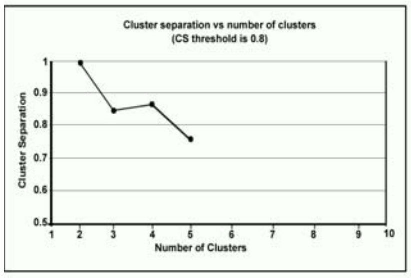


Figure 3: Number of Clusters vs. Cluster Separation

Our **DGEMSTNFMC** algorithm works in two phases. The outcome of the first phase (lines 1-19) of the **DGEMSTNFMC** algorithm consists of best number of noise-free clusters with their center. It first constructs **EMST1** form set of point S (line 1). Average weight of edges and standard deviation are computed (lines 2-3). Inconsistent edges are identified and removed from **EMST1** to generate subtree *T'* (lines 7-9). The center for each subtree (cluster/region) is computed at line 11. Using the cluster/region center point again another minimum spanning tree **EMST2** is constructed (line 13). Using the new evaluation criteria, best number of clusters/regions is identified (lines 14-16). Outliers are identified and removed from clusters (lines 17-18). Lines 6-18 in the algorithm are repeated until best or optimal number of noise-free clusters is obtained. We use the graph of Figure 4 as example to illustrate the second phase (lines 18-24) of **DGEMSTNFMC** algorithm.

The second phase of **DGEMSTNFMC** algorithm construct Minimum Spanning Tree *T* from the point set $C = \{c_1, c_2, c_3 \ldots c_k\}$ and convert the *T* into dendrogram is shown in figure 5. It places the entire disjoint cluster at level 0 (line 20). It then checks to see if *T still* contains some edge (line 21) If so, it finds minimum edge *e2* (line 22). It then finds the vertices *a, b* of an edge *e2* (line 23). It then merges the vertices (clusters) and forms a new vertex (*agglomerative approach*). At the same time the sequence number is increased by one and the level of the new cluster is set to the edge weight (line 24). Finally, the Updation of minimum spanning tree is performed at line 25. The lines 22-25 in the algorithm are repeated until the minimum spanning tree *T* has no edge. The dendrogram with best number of noise-free cluster as objects is generated (figure 5). The objects within the clusters are compact. The noise-free clusters are well separated, shown in Figure 4.

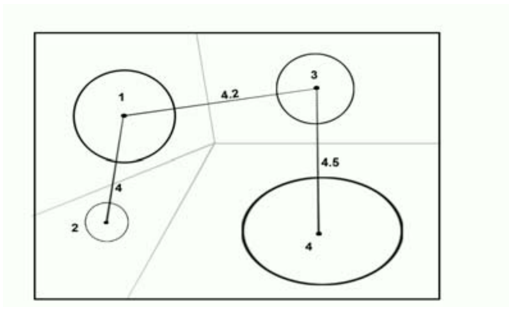


Figure 4. EMST2 From 4 region/cluster center points

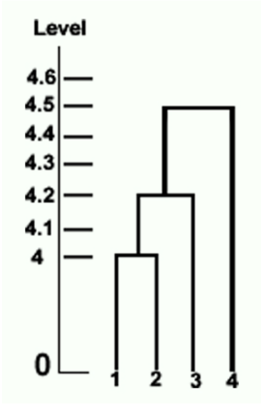


Figure 5. Dendrogram for Best number of noise-free Meta cluster

4. CONCLUSION

Our **DGEMSTNFMC** 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 **DGEMSTNFMC** clustering algorithm 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. Our algorithm also generates *dendrogram* which is used to find the relationship between the best numbers of clusters. The inter-cluster distances between clusters/regions are shown in the Dendrogram. This will be very useful in many applications. 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. The **DGEMSTNFMC** algorithm uses both divisive and agglomerative approach to find *Best number of Meta similarity noise-free clusters*. We will further study the rich properties of EMST-based clustering methods in solving different clustering problems.

REFERENCES

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

[2] 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.

[3] G. Williams, R. Baxter, H. He, S. Hawkins and L. Gu, "A Comparative Study for RNN for Outlier Detection in Data Mining", *In Proceedings of the 2nd IEEE International Conference on Data Mining*, page 709, Maebashi City, Japan, December 2002.

[4] T. Zhang, R. Ramakrishnan, and M. Livny. "BIRCH: A new data clustering algorithm and its applications". *Data Mining and Knowledge Discovery*, 1(2):141-182, 1997.

[5] C. Aggarwal and P. Yu, "Outlier Detection for High Dimensional Data". *In Proceedings of the ACM SIGMOD International Conference on Management of Data, Volume* 30, Issue 2,pages 37 – 46, May 2001.

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

[7] 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.

[8] 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.

[9] P. Hansen and M. Delattre, "Complete-link cluster analysis by graph coloring" *Journal of the American Statistical Association* 73, 397-403, 1978.

[10] 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.

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

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

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

[14] S. John Peter, S.P. Victor, "A Novel Algorithm for Meta similarity clusters using Minimum spanning tree". *International* Journal *of Computer Science and Network Security,* Vol.10 No.2 pp. 254 – 259, 2010

[15] 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.

[16] R. Prim. "Shortest connection networks and some generalization". *Bell systems Technical Journal,*36:1389-1401, 1957.

[17] J. Kruskal, "On the shortest spanning subtree and the travelling salesman problem", *In Proceedings of the American Mathematical Society,* pp 48-50, 1956.

[18] J. Nesetril, E.Milkova and H.Nesetrilova. Otakar boruvka on "Minimum spanning tree problem": Translation of both the 1926 papers, comments, history. DMATH*: Discrete Mathematics,* 233, 2001.

[19] 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.

[20] M. Fredman and D. Willard. "Trans-dichotomous algorithms for minimum spanning trees and shortest paths". In *Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science,*pages 719-725, 1990.

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

[22] C. Zahn. "Graph-theoretical methods for detecting and describing gestalt clusters". *IEEE Transactions on Computers*, C-20:68-86, 1971.

[23] F. Preparata and M.Shamos. "Computational Geometry": An Introduction*. Springer-Verlag, Newyr, NY ,USA*, 1985

[24] 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,*Pages 252-257, 1988.

[25] Y.Xu, V.Olman and D.Xu. "Minimum spanning trees for gene expression data clustering". *Genome Informatics,*12:24-33, 2001.

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

[27] 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.

[28] 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.

[29] 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.

[30] S.Papadimitriou, H.Kitawaga, P.Gibbons and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral" *Proc. Of the International Conference on Data Engineering* ,pp.315-326, 2003.

[31] 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.

[32] 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.

[33] J.Almeida, L.Barbosa, A.Pais and S.Formosinho, "Improving Hierarchical Cluster Analysis: A new method with OutlierDetection and Automatic Clustering," *Chemometrics and Intelligent Laboratory Systems* 87:208-217, 2007.

[34] K.Yoon, O.Kwon and D.Bae, "An approach to outlier Detection of Software Measurement Data using the K-means Clustering Method", *First International Symposium on Empirical Software Engineering and Measurement(ESEM 2007), Madrid.,*pp:443-445, 2007.

[35] D. Avis "Diameter partitioning." *Discrete and Computational Geometr,* 1:265-276, 1986.

[36] D. Johnson, "The np-completeness column: An ongoing guide". *Journal of Algorithms*,3:182-195, 1982.

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

[38] S. C. Johnson, "Hierarchical clustering schemes" *Psychometrika* 32, 241-254, 1967.

[39] Hubert L. J "Min and max hierarchical clustering using asymmetric similarity measures" *Psychometrika* 38, 63-72, 1973.

[40] J.C. Gower and G.J.S. Ross "Minimum Spanning trees and single-linkage cluster analysis" *Applied Statistics* 18, 54-64, 1969.

[41] 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.

[42] 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.

[43] 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.

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

[45] 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.

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

[47] 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,Vol20,no 16, pp2605-2617, 2004.