Outlier Removal Clustering through Minimum Spanning Tree

T. Karthikeyan
Department of Computer Science,
PSG College of Arts and Science,
Coimbatore,
Tamil Nadu, India

S. John Peter
Department of Computer
Science and Research Center
St. Xavier's College, Palayamkottai,
Tamil Nadu, India

ABSTRACT
Minimum spanning tree-based clustering algorithm is capable of detecting clusters with irregular boundaries. Detecting outliers using clustering algorithm is a big desire. Outlier detection is an extremely important task in a wide variety of application. In this paper we propose a minimum spanning tree-based clustering algorithm for detecting outliers. The algorithm partition the dataset into optimal number of clusters. Outliers are detected in the clusters based on outlyingness factor of each point (objects) in the cluster. 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. The algorithm works in two phases. The first phase of the algorithm creates optimal number of clusters, where as the second phase of the algorithm detect outliers.

Keywords:
Euclidean minimum spanning tree, Eccentricity, Cluster validity, Cluster Separation, Outlyingness factor, 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 [13]. 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. Erroneous outliers are observations that are distorted due to misreporting or misrecording errors in the data collection process. Outliers of either type may exert undue influence on the result of data analysis. So they should be identified using reliable detection methods prior to performing data analysis [13].

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 [1].

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 [3]. 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 [26, 29]. 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 [17]. 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 [19]. So, to improve the clustering such algorithm use the same process and functionality to solve both clustering and outlier discovery [9].

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 [32, 25, 28]. 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 [21, 11, 18]. 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.

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

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

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

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

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 [5, 40].

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 [14]. 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 [14] 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 [15]. 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 [20, 30] 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.

Our **ORCMST** clustering algorithm is based on Minimum Spanning Tree does not require a predefined cluster number as input. 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 optimal numbers of clusters are obtained. Using the optimal number of clusters outliers can be easily detected based on *outlyingness factor*. In section 2 we review some of the existing works on outlier detection, cluster validity and minimum spanning tree based clustering algorithms. In Section 3 we propose **ORCMST** algorithm which produces optimal number of clusters with outliers. Finally in conclusion we summarize the strength of our methods and possible improvements.

2. RELATED WORK

There is no single universally applicable or generic outlier detection approach [26, 29]. Therefore there are 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 [16, 7] 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 can not be applied in multidimensional dataset because of the univariate in nature. In addition, prior knowledge of the data distribution is required. These limitations have restricted the ability to apply these types of methods to large real-world databases which typically have many different fields and have no easy way of characterizing the multivariate distribution.

In the *distance-based approaches* [22, 23, 24, 33], 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 with in 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. Angiulli [4] propose a new definition of outliers. In this definition, for each point, *P*, the sum of the distances from its *k* nearest neighbor's considered. This sum is called the weight of *P*, W_k *(P)* and is used to rank the points of data set. Outliers are those points having the largest values of W_k. The method proposed by Angiulli [4] needs expected number of outlier n and application dependent k as input parameter. It is difficult to predict correct values for *n* and *k*. The problem with distance based approach is its high computational complexity.

Density-based approaches [8, 31] compute the density of the regions in the data and declare the objects in low dense regions as outliers. Clustering-based approaches [26, 12, 17, 19], 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.[19] 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 [26] 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 [2] is also used similar approach for detecting outliers. Using the K-means clustering algorithm Yoon [41] proposed a method to detect outliers. The K-means algorithm is sensitive to outliers, and hence it may not give accurate result.

Moh'd Belal Al-Zoubi [27] proposed a method based on clustering approaches for outlier detection using Partitioning Around Medoid (**PAM**). **PAM** attempts to determine *k* partition for *n* objects. The algorithm uses most centrally located object in a cluster (called medoid) instead of cluster mean. **PAM** is more robust than the k-means algorithm in the presence of noise and outlier. This **PAM** based approach suffer from proper cluster Structure. Cluster with irregular boundaries can not be detected using both k-means and **PAM** algorithms.

Clustering algorithm based on minimum and maximum spanning tree were extensively studied. Avis [6] found an $O(n^2 \log^2 n)$ algorithm for the min-max diameter-2 clustering problem. Asano, Bhattacharya, Keil and Yao [5] later 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 [5]. 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 [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. 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.

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 [35, 36, 37, 39]. Many of them based on the within and between-group distance.

3. ORCMST ALGORITHM FOR OUTLIERS

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 $W_e > \hat{W} + \sigma$ (or) *current longest edge* is removed from the tree. This leads to a set of disjoint subtrees $S_T = \{T_1, T_2 \ldots\}$. Each of these subtrees T_i is treated as cluster. We propose a new algorithm namely, *Outlier Removal Clustering through Minimum Spanning Tree* algorithm *(***ORCMST**), 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). The centers of clusters are identified using eccentricity of points. These points are a representative point for the each cluster or 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. Based on the definition of *small clusters* as defined in [26], we define *small cluster* as *a cluster with fewer points than half the average number of points in the optimal number of clusters.* We first detect small clusters (outliers) from optimal number of clusters. To detect the outliers from the rest of the clusters (if any), we propose a new method based on Minimum Spanning Tree.

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 **ORCMST** 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*:

$$R(p) = \{x \in D \mid dist(x, p) \leq dist(x, q) \ \forall q\} \qquad (1)$$

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 **DGMST** algorithm [10].

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

$$CS = E_{min} / E_{max} \qquad (2)$$

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[10]. Figure 5 shows the CS value versus the number of clusters in hierarchical clustering. The CS value < 0.8 when the number of clusters is 4. Thus, the proper number of clusters for the data set is 3. 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 **ORCMST** algorithm when it is used for clustering large dataset to detect outliers.

To detect the outliers from the rest of the clusters, we assign an *outlyingness factor* for each object in the subtree (cluster). Factor depends on its distance from cluster centroid. The method finds the objects with maximum distance to the partitioned centroid d_{max} is defined as:

$$d_{max}^{\ i} = max\{ ||dist(x_i^j - c_i)|| \} \qquad (3)$$

where the variable *i* refers the optimal number of clusters and *j* is used for the objects (points) in the i^{th} cluster(subtree/MST). *Outlyingness factor* for each point (objects) in the clusters

(subtrees/MST) are then computed. We define the *outlyingness* using Minimum Spanning Tree as:

$$O_i^j = dist(x_i^j - c_i) / d_{max}^i \qquad (4)$$

We see that all *outlyingness factor* of the data set are normalized to the scale [0, 1]. The greater the value, more likely the objects is an outlier. As an example of the data set clustered in to two clusters (shown in Figure 6 and 7) and computed *outlyingness factor* for the clusters are shown in Figure 6 and 7. The point (object) for which $O_i > THR$ are defined as outlier and removed from dataset. When scanning the cluster (**EMST),** the edges are ordered from smaller to larger lengths. Then we define the threshold as:

$$THR = min\ (L_i - L_{i-1}) * t \qquad (5)$$

Where L_i is smallest in the order and t ∈ [0, 1] is a user defined parameter.

Algorithm: ORCMST ( )
Input : *S* the point set
Output : optimal number of clusters with *O* the set of outliers

Let *e1* be an edge in the **EMST1** from *S*
Let *e2* be an edge in the **EMST2** 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
Let *O* be set of outliers

1. Construct an **EMST1** from *S*
2. Compute the average weight of Ŵ of all the Edges from **EMST1**
3. Compute standard deviation σ of the edges from **EMST1**
4. $S_T = ø$; $n_c = 1$; $C = ø$; $O = ø$;
5. **Repeat**
6. **For** each *e1* ∈ **EMST1**
7. **If** ($W_e > \hat{W} + \sigma$) or (current longest edge *e1)*
8. Remove *e1* from **EMST1**
9. $S_T = S_T \cup \{ T' \}$ // *T*'' is new disjoint subtree (cluster)
10. $n_c = n_c + 1$
11. Compute the center C_i of T_i using eccentricity of points
12. $C = U_{Ti} \in S_T \{C_i\}$
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. **Until** *CS < 0.8*
18. **For** each T_i (subtree/cluster) do
19. Determine *Small clusters* as outliers; *O = O U {small clusters}*
20. **For** each T_i (cluster) compute the d_{max}^i using equation (3)
21. **For** each point (objects) x_j in the cluster *i*,
22. Compute outlyingness O_j^i using equation(4)
23. **IF** $O_j^i > THR_i$ **then** remove the object x_j^i *;* $O = O \cup \{ x_j^i \}$
24. **Return** optimal number of noise-free clusters with outliers *O*

Figure 1 shows a typical example of **EMST1** constructed from point set S (from Dissimilarity Matrix), 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. The algorithm finds the center of the each cluster, which will be useful in many applications. The algorithm will find optimal number of clusters or cluster structures. Figure 2, 3 and 4 shows the possible distribution of the points in the cluster structures with their center vertices.

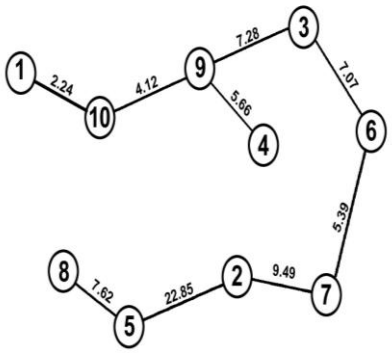


Figure 1: Clusters connected through points -EMST1

Our **OCRMST** algorithm works in two phases. The outcome of the first phase (lines 1-17) of the algorithm consists of optimal number clusters with their center. It first constructs **EMST1** from 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, optimal number of clusters/regions is identified (lines 14-16). Lines 6-16 in the algorithm are repeated until optimal number of clusters are obtained. Outliers which are present in the clusters are detected using the definition of the *small clusters* (line 18). We use the graph of figure 6 as example to illustrate the second phase (lines 20-24) of the algorithm.

The second phase of the **ORCMST** algorithm finds outliers from the optimal number of clusters, which are generated from first phase of the algorithm. The values of *outlyingness factor* and *THR* are calculated for each cluster (line 22). If the outliers are present in the cluster are detected based on the *outlyingness factor* and *THR* values (line 23) is shown in figure 6. The objects within the clusters are compact and noise-free. The clusters are well separated, shown in Figure 3.

3.2 Results and Discussion

Generally in most of the clustering algorithm data points can be represented as Dissimilarity Matrix (**DM**) representation. It contains the distance values between the data points represented as lower or upper triangular matrix. The distance calculation measure we used Euclidean distance between data points.

$$d(i, j) = \sqrt{(|X_{i,1} - X_{j,1}|^2 + |X_{i,2} - X_{j,2}|^2 + \ldots\ldots\ldots\ldots + |X_{i,n} - X_{j,n}|^2)} \qquad (6)$$

where i, j are n-dimensional data points.

Consider the sample data about the semester percentage marks of post graduate students as shown in Table 1.

TABLE 1: SAMPLE DATA

Student ID	Semester I Mark	Semester II Mark
1	65	70
2	62	65
3	72	74
4	61	80
5	71	86
6	73	67
7	71	62
8	78	83
9	65	76
10	64	72

The **DM** for the above sample data is shown in Table 2.

TABLE 2: DISSIMILARITY MATRIX

		2	3	4	5	6	7	8	9	10
1	0	5.83	8.06	10.77	17.09	8.54	10.00	18.38	6.00	2.24
2		0	13.45	15.03	22.85	11.18	9.49	24.08	11.40	7.28
3			0	12.53	12.04	7.07	12.04	10.82	7.28	8.25
4				0	11.66	17.69	20.59	17.26	5.66	8.54
5					0	19.10	24.00	7.62	11.66	15.65
6						0	5.39	16.76	12.04	10.30
7							0	22.14	15.23	12.21
8								0	14.76	17.80
9									0	4.12
10										0

TABLE 3: MINIMUM SPANNING TREE EDGES

Edge	**Euclidean Distance/Weight**
{1,10}	2.24
{10,9}	4.12
{9,4}	5.66
{9,3}	12.04
{3,6}	7.07
{6,7}	5.39
{7,2}	9.49
{2,5}	22.85
{5,8}	7.62

Our **DHCMST** algorithm constructs **EMST1** from the dissimilarity matrix is shown in the figure 1. The mean $\hat{W}$ and standard deviation σ of the edges from the **EMST1** are computed respectively as 7.986 and 5.966. The sum of the mean $\hat{W}$ and standard σ is computed as 13.934. This value is used to identify the inconsistence edges in the **EMST1** to generate subtree (clusters). Based on the above value the edge having weight 22.85 connecting the vertices 2 and 5 is find to be inconsistent one. By removing the inconsistent edge from the **EMST1**, vertices (data points) in the **EMST1** partitioned into two sets (two subtrees or clusters) T_1 and T_2 namely T_1 = *{1, 10, 9, 4, 3, 6, 7, 2}* and T_2 = *{5, 8} is* shown in the figure 2. Center point (vertex) for each of the each subtree is find using eccentricity of points (vertices). These center point or vertex is connected and again another Minimum Spanning Tree **EMST2 (**figure 2**)** is constructed. Using the **EMST2** the minimum edge (E_{min} = 19.1) and maximum edge (E_{max} = 19.1) is find to compute Cluster Separation value (CS=1). If the CS is greater than 0.8, then we conclude the subtrees (clusters) created are well separated. Next many inconsistency edges are identified based on the inconsistency condition to create subtrees (clusters). Center point for each of the subtree or cluster is finding using eccentricity of points. By connecting the center points (vertices) again **EMST2** is constructed. Using the **EMST2** again new CS value is computed. The process is repeated until to create optimal number of clusters is shown in the figure 3. For the given data the above process terminates when E_{min} = 7.62, E_{max} = 13.45 and CS = 0.566 is shown in the figure 4. Our **ORCMST** algorithm creates three well separated disjoint subtrees (clusters) for the given data, namely T_1 = *{5, 8}*, T_2 = *{1, 10, 9, 4, 3, 6, 7}* and T_3 = *{2}*. Each of the subtree T_i is considered as clusters. From the outcome of the clustering it is clear that the students having id number 5 and 8 are very good in the first two semester examination, where as the student id number with 2 is average in the semester examinations. The remaining students are neither too good nor average in the two semester examination, but they are good in the semester examinations. Based on the definition of *small clusters* vertex 2 is termed as outlier.

The second phase the **ORCMST** algorithm is illustrated with the help of figure 6. Using the **EMST** eccentricities of all the points are computed. The value of d_{max} using equation (3) is computed as 34.93. The center point or vertex (vertex 9) of the Minimum Spanning Tree **EMST** is identified. Then *outlyingness factor* for each data point in **EMST** is computed using equation (4) is shown in figure 6. This value is used to identify the outliers in the data set. The point or vertices 3 and 6 (*outlyingness factor* value 0.93 and 1) are far away from the center of the **EMST** (vertex 9) is considered as outliers.

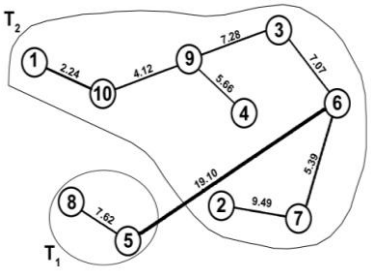


Figure 2: EMST2 from Two Clusters using Center points 5 and 6; E_{min} , E_{max} = 19.1 and CS =1

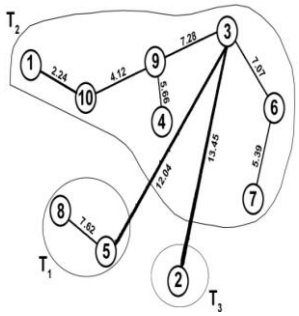


Figure 3: EMST2 from Three Clusters using Center points 5, 3 and 2;E_{min} =12.04 ,
E_{max} = 13.45and CS = 0.895

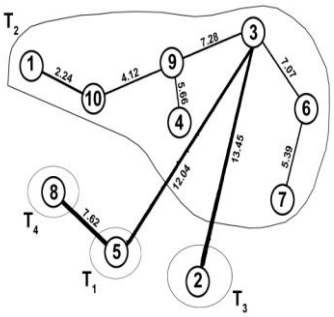


Figure 4: EMST2 from 4 cluster center points 8, 5, 3 and 2; E_{min} =7.62, E_{max} = 13.45 and CS = 0.566

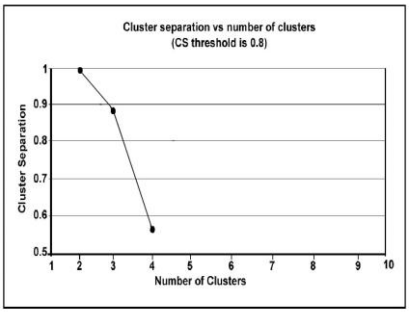
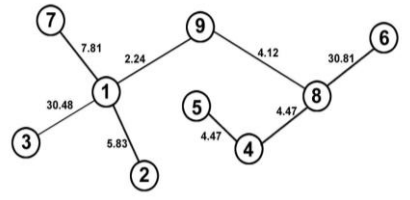


Figure 5: No of Clusters Vs Clusters Separation



Vertex v	1	2	3	4	5	6	7	8	9
Eccen (V)	37.17	43.0	67.65	41.31	45.78	67.65	44.98	36.84	34.93
Dist (V,C(T))	2.24	8.07	32.72	8.59	13.06	34.93	10.05	4.12	0
Dist (V,C(T)) / d_{max}	0.06	0.23	0.93	0.24	0.37	1	0.29	0.11	0

Center Vertex : 9
d_{max} : 34.93

Figure 6: Outlyingness factor for objects from cluster (Subtree); Vertices 3 and 6 are outliers

4. CONCLUSION

In this paper we have proposed a new method to integrate outlier removal into Minimum Spanning Tree based Clustering algorithm. Our **ORCMST** clustering algorithm finds outliers and clusters without using any predefined cluster number as input parameter. Our algorithm does not require the users to select and try various parameters combinations in order to get the desired output. Our **ORCMST** clustering algorithm uses a new cluster validation criterion based on the geometric property of partitioned clusters to produce optimal number of "true" clusters with outliers for each of them. The proposed algorithm was tested using two different types of synthetic data sets. The test result shows that the proposed method outperformed the baseline methods. In 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 for detecting outliers in dynamic data set.

5. REFERENCE

[1] 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, pp 37 – 46, May 2001.

[2] 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.

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

[4] F.Angiulli, and C.Pizzuti, Outlier Mining in Large High-Dimensional Data sets,.IEEE Transactions on Kaoeledge and Data Engineering, 17(2): 203-215, 2005.

[5] 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 ComputationalGeometry*,Pages252-257,1988.

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

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

[8] 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.

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

[10] 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.

[11] 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.

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

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

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

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

[17] 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.

[18] 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.

[19] 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.

[20] 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

[21] 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.

[22] 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.

[23] 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.

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

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

[26] 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.

[27] Moh'd Belal Al-Zoubi, "An Effective Clustering-Based Approach for Outlier Detection", *European Journal of Scientific Research*, Vol.28 No.2 ,pp.310-316, 2009

[28] 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.

[29] 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.

[30] 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.

[31] 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.

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

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

[34] R. Rezaee, B.P.F. Lelie and J.H.C. Reiber, "A new cluster validity index for the fuzzy C-mean", *Pattern Recog. Lett., 19*,237-246, 1998.

[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] 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.

[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.