Clustering the Labeled and Unlabeled Datasets using New MST based Divide and Conquer Technique

Srinivasulu Asadi, Dr.Ch.D.V.Subba Rao, V.Saikrishna and Bhudevi Aasadi

Abstract: **Clustering is the process of partitioning the data set into subsets called clusters, so that the data in each subset share some properties in common. Clustering is an important tool to explore the hidden structures of modern large Databases. Because of the huge variety of the problems and data distributions, different classical clustering algorithms, such as hierarchical, partitional, density-based and model-based clustering approaches, have been developed and no techniques are completely satisfactory for all the cases. Sufficient empirical evidences have shown that a New Minimum Spanning Tree (NMST) representation is quite invariant to the detailed geometric changes in cluster boundaries. Therefore, the shape of a cluster has little impact on the performance of MST - based clustering algorithms, which allows us to overcome many of the problems faced by the classical clustering algorithms. NMST - based clustering algorithms also have the ability to detect clusters with irregular boundaries and so they are being widely used in practice. In these MST - based clustering algorithms, search for nearest neighbour is to be done in the construction of NMST. This search is the main source of computation and the standard solutions take O(N^2) time. In our paper, we present a fast minimum spanning tree-inspired clustering algorithm. This algorithm uses an efficient implementation of the cut and the cycle property of the NMST, that can have much better performance than O(N^2) time.**

General Terms: **Data Mining, Image Processing, Artificial Intelligence, Graph Theory and Design Analysis of Algorithms.**

Keywords **— Clustering, Minimum Spanning Tree, Kruskal's algorithm, Prim's algorithm , K-Means Clustering, C-Means Clustering, Divisive Hierarchical Clustering.**

1. INTRODUCTION

For a set of data points and a distance measure, clustering is the process of partitioning the data set into subsets, called clusters, so that the data in each subset share some properties in common. Usually, the common properties are quantitatively evaluated by some measures of the optimality such as minimum intracluster distance or maximum intercluster distance, etc. Clustering, is an important tool to explore the hidden structures of modern large databases, has been extensively studied and many algorithms have been proposed in the literature. Because of the huge variety of the problems and data distributions, different techniques, such as hierarchical, partitional, density and model-based approaches, have been developed and no techniques are completely satisfactory for all the cases. For example, some classical algorithms rely on either the idea of grouping the data points around some "centres" or the idea of separating the data points using some regular geometric curves such as hyper planes. As a result, they generally do not work well when the boundaries of the clusters are irregular.

MST is only one of several spanning tree problems that arise in practice. There are other spanning tree-based clustering algorithms that maximize or minimize the degrees of link of the vertices. However, these algorithms are computationally expensive. MST-based clustering algorithms have been studied for decades. With the coming of information explosion, computational efficiency has become a major issue for modern large databases which typically consist of millions of data items. As random access memory is getting cheaper, larger and larger main memories become possible to store the whole database for faster system response. As a result, very efficient MST-based in-memory clustering algorithms are in need. In the past, k-d tree (for nearest neighbour search to avoid some distance computation) and the Delaunay Triangulation had been employed in the construction of MST to reduce the time complexity to near O(N log N). Unfortunately, they work well only for dimensions no more than 5. Although many new index structures for nearest neighbour search in high-dimensional databases have been proposed recently, their applications to the MST problem have not been reported. In our paper, we propose a new MST - inspired clustering approach that is both computationally efficient and competent with the state-of-the-art MST - based clustering techniques. Basically, our MST - inspired clustering technique tries to identify the relatively small number of inconsistent edges and remove them to form clusters before the complete MST is constructed. To be as general as possible, our algorithm has no specific requirements on the dimensionality of the data sets and the format of the distance measure, though Euclidean distance is used as the edge weight in our experiments.

2. REVIEW OF MINIMUM SPANNING TREE ALGORITHM

In traditional MST problems, a set of n vertices and a set of m edges in a connected graph are given. A "generic" minimum spanning tree algorithm grows the tree by adding one edge at a time. Two popular ways to implement the generic algorithm are the Kruskal's algorithm and the Prim's algorithm. In opposition to the "generic" minimum spanning tree algorithms, "Reverse Delete" algorithm starts with the full graph and deletes edges in order of non increasing weights based on the cycle property as long as it does not disconnect the graph. The cost of constructing an MST using these classical MST algorithms is O (m log n).

2.1 MST-BASED CLUSTERING ALGORITHMS

With an MST being constructed, the next step is to define an edge inconsistency measure so as to partition the tree into clusters. Like many other clustering algorithms, the number of clusters is either given as an input parameter or figured out by the algorithms themselves. Under the ideal condition, that is, the clusters are well separated and there exist no outliers, the inconsistent edges are just the longest edges. However, in real-world tasks, outliers often exist, which make the longest edges an unreliable indication of cluster separations.

In these cases, all the edges that satisfy the inconsistency measure are removed and the data points in the smallest clusters are regarded as outliers. As a result, the definition of the inconsistent edges and the development of the terminating condition are two major issues that have to be addressed in all MST-based clustering algorithms, even when the number of clusters is given as an input parameter. Due to the invisibility of the MST representation of a data set of dimensionalities beyond 3, many inconsistency measures have been suggested in the literature.

Clustering is the process of partitioning the data into subsets, called clusters. Because of huge variety of problems different techniques such as hierarchical, density-and model-based approaches are not satisfactory. The problems faced by classical clustering algorithms can be reduced by Minimum spanning tree representation up to some extent. A "generic" minimum spanning tree algorithm grows the tree by adding one edge at a time .Two popular ways to implement the generic algorithm are the Kruskal's algorithm and the Prim's algorithm.

In the Kruskal's algorithm, all the edges are sorted into a non decreasing order by their weights, and the construction of an MST starts with n trees, i.e., every vertex being its own tree. Then for each edge to be added in such a non decreasing order, check whether its two endpoints belong to the same tree. If they do (i.e., a cycle will be created), such an edge should be discarded. In the Prim's algorithm, the construction of an MST starts with some root node t and the tree T greedily grows from t outward. At each step, among all the edges between the nodes in the tree T and those not in the tree yet, the node and the edge associated with the smallest weight to the tree T are added.

In opposition to the "generic" minimum spanning tree algorithms, "Reverse Delete" algorithm starts with the full graph and deletes edges in order of non increasing weights based on the cycle property as long as doing so does not disconnect the graph. The cost of constructing an MST using these classical MST algorithms is O(m log n). More efficient algorithms promise close to linear time complexity under different assumptions. In an MST-based clustering algorithm, the inputs are a set of N data points and a distance measure defined upon them. Since every pair of points in the point set is associated with an edge, there are $N(N - 1)/2$ such edges. The time complexity of the Kruskal's algorithm, the Prim's algorithm, and the "Reverse Delete" algorithm adapted for this case is $O(N^2)$.

With an MST being constructed, the next step is to define an edge inconsistency measure so as to partition the tree into clusters. Like many other clustering algorithms, the number of clusters is either given as an input parameter or figured out by the algorithms themselves. Under the ideal condition, that is, the clusters are well separated and there exist no outliers, the inconsistent edges are just the longest edges. However, in real-world tasks, outliers often exist, which make the longest edges an unreliable indication of cluster separations. In these cases, all the edges that satisfy the inconsistency measure are removed and the data points in the smallest clusters are regarded as outliers.

As a result, the definition of the inconsistent edges and the development of the terminating condition are two major issues that have to be addressed in all MST-based clustering algorithms, even when the number of clusters is given as an input parameter. Due to the invisibility of the MST representation of a data set of dimensionalities beyond 3, many inconsistency measures have been suggested in the literature. In Zahn's original work, the inconsistent edges are defined to be those whose weights are significantly larger than the average weight of the nearby edges in the tree. The performance of this clustering algorithm is affected by the size of the nearby neighborhood.

3. DIVISIVE HIERARCHICAL CLUSTERING ALGORITHM (DHCA)

Essentially, for a given data set and "K" value, the DHCA start with K randomly selected centres, and then assigns each data point to its closest centre, creating K partitions. At each stage in the iteration, for each of these K partitions, DHCA recursively selects K random centres and continues the clustering process within each partition to form at most K^N partitions for the N^{th} stage[1]. In our implementation, the procedure continues until the number of elements in a partition is below K+2, at which time, the distance of each data item to other data items in that partition can be updated with a smaller value by a brute-force nearest neighbour search.The Divisive Hierarchical Clustering Algorithm partitions the data set into smaller partitions so that the number of data items in each partition must be less than the maximum partition size i.e., "K+2". In the first iteration the entire data set is stored as the initial partition. After that, at each stage all the partitions are stored irrespective of their "K+2" condition.

For a set of s-dimensional data, i.e., each data item is a point in the s-dimensional space, there exists a distance between every pair of the data items. In this sequential initialization, all the pair wise distances are calculated after reading their details from the database. The threshold value calculation consists of Distance and Index Arrays. The distance array[1] is used to record the distance of each data point to some other data point in the sequentially stored data set. The index array[1] records the index of the data item at the other end of the distance in the distance array. The number of partition centers at each stage of the DHCA, performance of the algorithm with and without a given number of clusters. K varies from min to max. For max k, the algorithm produces into larger number of distance computations in the DHCA processes and the running time increases with k. As K increases, the change in running time increases along. For minimum value of K, the changes in the running time are small because when K is minimum for the construction of DHCA a small increase in k, decreases the total number of nodes which makes better in performance than using small k. When K gets larger and larger, more distance processes to partition center and the increase in processes eventually improves.

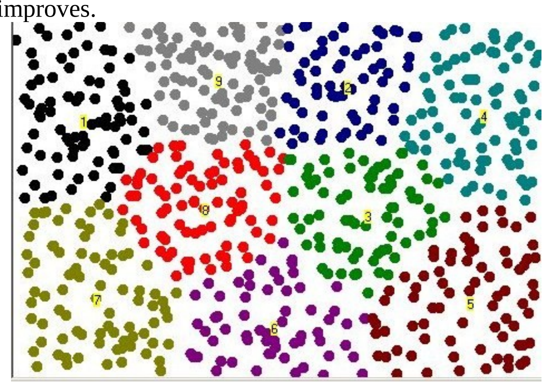


Fig 1. Random Selection of Centers and nearest neighbor assignment in DHCA

3.1. Grygorash MST-based clustering techniques

At almost the same time, Grygorash proposed two MST-based clustering algorithms, called the Hierarchical Euclidean distance - based MST clustering algorithm (**HEMST**) and the Maximum Standard Deviation Reduction clustering algorithm (**MSDR**) respectively. Requiring the number of clusters given as an input, their HEMST first computes the average and the standard deviation of the edge weights in the entire EMST and uses their sum as a threshold. Next, edges with a weight larger than the threshold are removed, leading to a set of disjoint sub trees. For each cluster, a representative is identified as its centroid, resulting in a reduced data set.

An EMST is next constructed on these representatives and the same tree partitioning procedure is followed until the number of clusters is equal to the preset number of clusters. With no input about the number of clusters, their MSDR is actually a recursive two partition optimization problem. In each step, it removes an edge only when the overall clusters weight standard deviation reduction is maximized. This process continues until such reduction is within a threshold and the desired number of clusters is obtained by finding the local minimum of the standard deviation reduction function. Since every edge in the tree is checked before a cutting, the problem with MSDR is its high computational cost, particularly for very large data sets.

3.1. Limitations of Existing System

The limitations of our existing system are:

- Inefficient for dimensionality more than 5.
- Huge number of computations.
- Do not work if the boundaries of clusters are irregular.
- More time complexity. Their standard solutions take about O(N^2) time complexity.

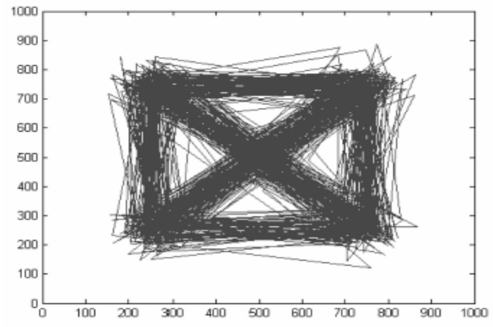


Fig. 2. Its spanning tree after the sequential initialization.

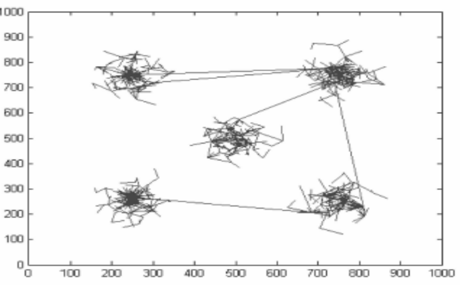


Fig. 4. Updated spanning tree using DHCAs.

Fig 3. Spanning Tree after SI and after DHCA updating respectively

In case, the data point is equidistant to two or more centres, the partition to which the data point belongs is the first one [1]. However, because any data point in a partition is closer to its cluster center (not its nearest neighbour) than to the center of any other partition, the data points in the clusters' boundaries can be misclassified into a wrong partition. Fortunately, such possibilities can be greatly reduced by multiple runs of DHCA. To summarize, the advantage of DHCA is that, after multiple runs, each point will be very close to its true nearest neighbour in the data set.

3.2 Marked Divisive Hierarchical Clustering Algorithm (MDHCA)

The MST-based clustering algorithms can be more efficient if the longest edges of an MST can be identified quickly before most of the shorter ones are found. This is because, for some MST-based clustering problems, if we can find the longest edges in the MST very quickly, there is no need to compute the exact distance values associated with the shorter ones. Second, for other MST-based clustering algorithms, if the longest edges can be found quickly, the Prim's algorithm can be more efficiently applied to each individual size-reduced cluster. For the cases where the number of the longest edges that separate the potential clusters can be much fewer than the number of the shorter edges this divide-and-conquer approach will allow us to save the number of distance computations tremendously.

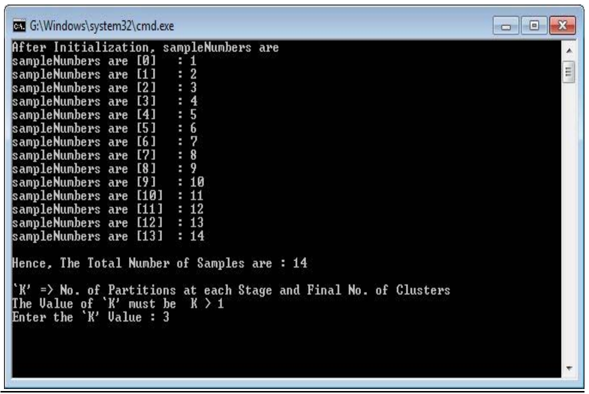
After the DHCA iterations, the partitions which are having data points strictly less than K+2 are marked. All these marked partitions are connected by searching their nearest neighbor in other partitions and connecting to them. This process is repeated until a MST is constructed. Now this MST is given as input to MDHCA. Here in this MDHCA, we use a flag array to mark all the points on one side of the longest edge to be 1 and all the points on the other side to be 0. Then the DHCA can be applied multiple times with the partition centers being chosen only from the data points marked either 1 or 0, but not both. We call this procedure the marked DHCA (MDHCA).

4. NMST-INSPIRED CLUSTERING ALGORITHM

In this MST-Inspired clustering algorithm, both the DHCA and MDHCA are performed in a combined manner in this algorithm. The input is S - Dimensional Data Set[1] and K for the DHCA and MDHCA, while the output will be a Labeling array that remembers the cluster label each data item belongs to. MST-inspired clustering algorithm can be summarized in the following steps:

Step (1) : Start with a spanning tree built by the Sequential initialization.
Step (2) : Calculate mean and standard deviation of the edge weights distance array.
Step (3) : Use their sum as the threshold.
Step (4) : Perform multiple runs of DHCA Algorithm.
Step (5) : Identify longest edge using MDHCA.
Step (6) : Remove this longest edge.
Step (7) : Check Terminating Condition and continue.
Step (8) : Put that number of clusters into C-Means clustering.

To summarize, the numerical parameters the algorithm needs from the user include the data set, the loosely estimated minimum and maximum numbers of data points in each cluster, the input "K" to the DHCA and MDHCA, and number of nearest neighbours to keep for each data item in the auxiliary arrays, while the outputs will be the final distance and index arrays, and a labelling array that remembers the cluster label each data item belongs to.

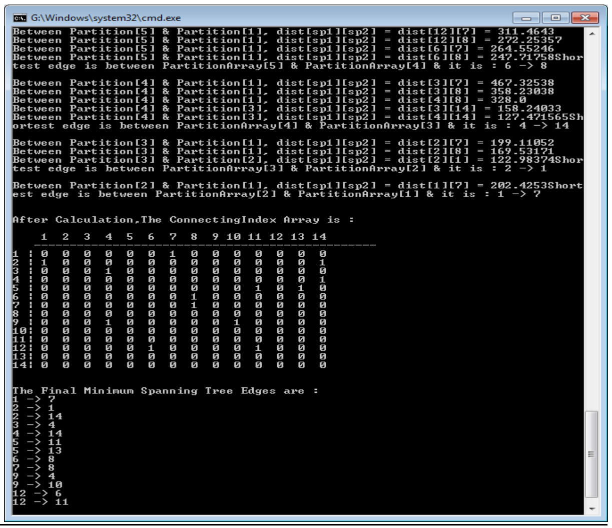


```
G:\Windows\system32\cmd.exe
After Initialization, sampleNumbers are
sampleNumbers are [0]   : 1
sampleNumbers are [1]   : 2
sampleNumbers are [2]   : 3
sampleNumbers are [3]   : 4
sampleNumbers are [4]   : 5
sampleNumbers are [5]   : 6
sampleNumbers are [6]   : 7
sampleNumbers are [7]   : 8
sampleNumbers are [8]   : 9
sampleNumbers are [9]   : 10
sampleNumbers are [10]  : 11
sampleNumbers are [11]  : 12
sampleNumbers are [12]  : 13
sampleNumbers are [13]  : 14

Hence, The Total Number of Samples are : 14

'K' => No. of Partitions at each Stage and Final No. of Clusters
The Value of 'K' must be  K > 1
Enter the 'K' Value : 3
```

Fig 4. Initialization and Input



```
G:\Windows\system32\cmd.exe
Between Partition[5] & Partition[1], dist[sp1][sp2] = dist[12][7] = 311.4643
Between Partition[5] & Partition[1], dist[sp1][sp2] = dist[12][8] = 272.25357
Between Partition[5] & Partition[1], dist[sp1][sp2] = dist[6][7] = 264.55246
Between Partition[5] & Partition[1], dist[sp1][sp2] = dist[6][8] = 247.71758Shor
test edge is between PartitionArray[5] & PartitionArray[4] & it is : 6 -> 8

Between Partition[4] & Partition[1], dist[sp1][sp2] = dist[3][7] = 467.32538
Between Partition[4] & Partition[1], dist[sp1][sp2] = dist[3][8] = 358.23038
Between Partition[4] & Partition[1], dist[sp1][sp2] = dist[4][8] = 328.0
Between Partition[4] & Partition[3], dist[sp1][sp2] = dist[3][14] = 158.24033
Between Partition[4] & Partition[3], dist[sp1][sp2] = dist[4][14] = 127.471565Sh
ortest edge is between PartitionArray[4] & PartitionArray[3] & it is : 4 -> 14

Between Partition[3] & Partition[1], dist[sp1][sp2] = dist[2][7] = 199.11052
Between Partition[3] & Partition[1], dist[sp1][sp2] = dist[2][8] = 169.53171
Between Partition[3] & Partition[2], dist[sp1][sp2] = dist[2][1] = 122.98374Shor
test edge is between PartitionArray[3] & PartitionArray[2] & it is : 2 -> 1

Between Partition[2] & Partition[1], dist[sp1][sp2] = dist[1][7] = 202.4253Short
est edge is between PartitionArray[2] & PartitionArray[1] & it is : 1 -> 7


After Calculation,The ConnectingIndex Array is :

    1  2  3  4  5  6  7  8  9 10 11 12 13 14
   -----------------------------------------
1 | 0  0  0  0  0  0  1  0  0  0  0  0  0  0
2 | 1  0  0  0  0  0  0  0  0  0  0  0  0  1
3 | 0  0  0  1  0  0  0  0  0  0  0  0  0  0
4 | 0  0  0  0  0  0  0  0  0  0  0  0  0  1
5 | 0  0  0  0  0  0  0  0  0  0  1  0  1  0
6 | 0  0  0  0  0  0  0  1  0  0  0  0  0  0
7 | 0  0  0  0  0  0  0  1  0  0  0  0  0  0
8 | 0  0  0  0  0  0  0  0  0  0  0  0  0  0
9 | 0  0  0  1  0  0  0  0  0  1  0  0  0  0
10| 0  0  0  0  0  0  0  0  0  0  0  0  0  0
11| 0  0  0  0  0  0  0  0  0  0  0  0  0  0
12| 0  0  0  0  0  1  0  0  0  0  1  0  0  0
13| 0  0  0  0  0  0  0  0  0  0  0  0  0  0
14| 0  0  0  0  0  0  0  0  0  0  0  0  0  0


The Final Minimum Spanning Tree Edges are :
1 -> 7
2 -> 1
2 -> 14
3 -> 4
4 -> 14
5 -> 11
5 -> 13
6 -> 8
7 -> 8
9 -> 4
9 -> 10
12 -> 6
12 -> 11
```

Fig 5. Final MST Edges Path

4.1 Advantages of Proposed System

- Efficient use of the cut and cycle properties by our Fast MST-Inspired clustering algorithm.
- Shape of a cluster has very little impact on the performance of this MST - based clustering algorithm.
- Efficient for dimensionality more than 5 and reduced time complexity.
- Nearest neighbor search is used to construct efficient MST.
- Works efficiently even if the boundaries of clusters are irregular.

5. CONCLUSION

The main objective of the paper is to perform Cluster Analysis in Data Mining very efficiently & effectively with reduced time comlexity. Our paper mainly deals with efficient Clustering of a Data Set. Our contribution is the design of a new MST-inspired clustering algorithm for large data sets by utilizing the DHCA in an efficient implementation of the cut and the cycle property which is a more efficient method that can quickly identify the longest edges in an MST so as to save some computations. Our system is efficient for any number of Dimensions and reduces Time Complexity. Also Irregular boundaries can be efficiently handled using our MST based Clustering using Divide and Conquer Technique. The future work is that we can do a further study of the rich properties of the existing MST algorithms and adapt our proposed NMST-inspired clustering algorithm to more general and larger data sets, particularly when the whole data set cannot fit into the main memory.

6. REFERENCES

[1] Xiaochun Wang, Xiali Wang and D. Mitchell Wilkes, IEEE Members, "A Divide-and-Conquer Approach for Minimum Spanning Tree-Based Clustering", IEEE Knowledge and Data Engineering Transactions, vol 21, July 2009.

[2] Jiawei Han and Micheline Kamber. "Data Ware Housing and Data Mining. Concepts and Techniques", Third Edition 2007.

[3] N. Chowdbury and C.A. Murthy, "Minimum Spanning Tree-Based Clustering –Technique: Relationship with Bayes Classifier," Pattern Recognition, vol. 30.

[4] M. Laszlo and S. Mukherjee, "Minimum Spanning Tree Partitioning Algorithm for Micro aggregation," IEEE Trans. Knowledge and Data Engineering, vol. 17.

[5] I. Katriel, P. Sanders, and J.L. Traff, "A Practical Minimum Spanning Tree Algorithm Using the Cycle Property," Proc. 11th European Symp. Algorithms (ESA '03), vol. 2832, pp. 679-690, 2003.

[6] C.T. Zahn, "Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters," IEEE Trans. Computers, vol. 20, no. 1, pp. 68-86, Jan. 1971.

[7] D.J. States, N.L. Harris, and L. Hunter, "Computationally Efficient Cluster Representation in Molecular Sequence Megaclassification," ISMB, vol. 1, pp. 387-394, 1993.

[8] P. Raghu Rama Krishnan, "Accessing Databases with JDBC and ODBC", Pearson Education, Fourth Edition.

[9] Henry F Korth, S. Sudharshan, "Database System Concepts" McGraw – Hill,International Editions, Fourth Edition, 2002.

[10] I. Katriel, P. Sanders, and J.L. Traff, "A Practical Minimum Spanning Tree Algorithm Using the Cycle Property,"

[11] Sartaj Sahni, Advanced Data Structures and Algorithms, Second Edition.

[12] Grady Booch, James Rumbaugh, Ivar Jacobson. Unified Modeling Language User Guide. Pearson Education, 2006 edition.

[13] I. Katriel, P. Sanders, and J.L. Traff, "A Practical Minimum Spanning Tree Algorithm Using the Cycle Property," Proc. 11th European Symp. Algorithms (ESA '03), vol. 2832, pp. 679-690, 2003.

[14] C.T. Zahn, "Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters," IEEE Trans. Computers, vol. 20,no. 1, pp. 68-86, Jan. 1971.

[15] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis. Wiley-Interscience, 1973.

[16] N. Chowdhury and C.A. Murthy, "Minimum Spanning Tree- Based Clustering Technique: Relationship with Bayes Classifier," Pattern Recognition, vol. 30, no. 11, pp. 1919-1929, 1997.

[17] A. Vathy-Fogarassy, A. Kiss, and J. Abonyi, "Hybrid Minimal Spanning Tree and Mixture of Gaussians Based Clustering Algorithm," Foundations of Information and Knowledge Systems, pp. 313-330, Springer, 2006.

[18] O. Grygorash, Y. Zhou, and Z. Jorgensen, "Minimum Spanning Tree-Based Clustering Algorithms," Proc. IEEE Int'l Conf. Tools with Artificial Intelligence, pp. 73-81, 2006.

[19] R.C. Gonzalez and P. Wintz, Digital Image Processing, second ed. Addison-Wesley, 1987.

[20] Y. Xu, V. Olman, and E.C. Uberbacher, "A Segmentation Algorithm for Noisy Images: Design and Evaluation," Pattern Recognition Letters, vol. 19, pp. 1213-1224, 1998.

[21] Y. Xu and E.C. Uberbacher, "2D Image Segmentation Using Minimum Spanning Trees," Image and Vision Computing, vol. 15, pp. 47-57, 1997.

[22] D.J. States, N.L. Harris, and L. Hunter, "Computationally Efficient Cluster Representation in Molecular Sequence Megaclassification," ISMB, vol. 1, pp. 387-394, 1993.

[23] Y. Xu, V. Olman, and D. Xu, "Clustering Gene Expression Data Using a Graph-Theoretic Approach: An Application of Minimum Spanning Trees," Bioinformatics, vol. 18, no. 4, pp. 536-545, 2002.
[24] M. Laszlo and S. Mukherjee, "Minimum Spanning Tree Partitioning Algorithm for Microaggregation," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 7, pp. 902-911, July 2005.
[25] M. Forina, M.C.C. Oliveros, C. Casolino, and M. Casale, "Minimum Spanning Tree: Ordering Edges to Identify Clustering Structure," Analytical Chimica Acta, vol. 515, pp. 43-53, 2004.

AUTHORS BIOGRAPHY

Asadi Srinivasulu received the B Tech (CSE) from Sri Venkateswara University, Tirupati, India in 2000 and M.Tech with Intelligent Systems in IT from Indian Institute of Information Technology, Allahabad (IIIT) in 2004 and he is pursuing Ph.D in CSE from J.N.T.U.A, Anantapur, India. He has got 10 years of teaching and industrial experience. He served as the Head, Dept of Information Technology, S V College of Engineering, Karakambadi, Tirupati, India during 2007-2009. His areas of interests include Data Mining and Data warehousing, Intelligent Systems, Image Processing, Pattern Recognition, Machine Vision Processing and Cloud Computing. He is a member of IAENG, IACSIT. He has published more than 10 papers in International journals and conferences. Some of his publications appear in IJCA and IJCSIT digital libraries. He visited Malaysia and Singapore.

Dr Ch D V Subba Rao received the B Tech (CSE) from S V University College of Engineering, Tirupati, India in 1991, M.E. (CSE) from M K University, Madurai in 1998 and he was the first Ph.D awardee in CSE from S V University, Tirupati in 2008. He has got 19 years of teaching experience. He served as the Head, Dept of Computer Science and Engineering, S V University College of Engineering, Tirupati, India during 2008-11. His areas of interests include Distributed Systems, Advanced Operating Systems and Advanced Computing. He is a member of IETE, IAENG, CSI and ISTE. He chaired and served as reviewer of IAENG and IASTED international conferences. He has published more than 25 papers in International journals and conferences. Some of his publications appear in IEEE and ACM digital libraries. He visited Austria, Netherlands, Belgium, Hong-Kong, Thailand and Germany.

V.Saikrishna received B.Tech degree in Computer Science and Engineering from JNTUH, Hyderabad in 2004 and M.Tech degree in Computer Science from S.V.U, Tirupathi in 2008. He is pursuing Ph.D in SV University Tirupati. He has 5 years of experience in teaching. Currently working on Data warehousing Data mining recent trends in Computer Centre Laboratory at S.V.U.College of Engineering, his research areas are Databases and Data Mining.

Aasadi Bhudevi received B.Tech degree in Information Technology from JNTUA, Anantapur in 2011, her research areas are Data warehousing and Data Mining, Database Management Systems and Software Engineering.