ISSN No. 0976-5697

Volume 3, No. 1, Jan-Feb 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

New Density-Based Clustering Technique: GMDBSCAN-UR

Mohammed A. Alhanjouri*
Computer Engineering Department
Islamic University of Gaza (IUG) Gaza, Palestine
mhanjouri@iugaza.edu.ps

Rwand D. Ahmed
Computer Engineering Department
Islamic University of Gaza (IUG) Gaza, Palestine
rwando@iugaza.edu.ps

Abstract: Density Based Spatial Clustering of Applications of Noise (DBSCAN) is one of the most popular algorithms for cluster analysis. It can discover clusters with arbitrary shape and separate noises. But this algorithm cannot choose its parameter according to distribution of dataset. It simply uses the global minimum number of points (MinPts) parameter, so that the clustering result of multi-density database is inaccurate. In addition, when it used to cluster large databases, it will cost too much time. In this paper we try to solve these problems by integrated the grid-based in addition to using representative points in our new proposed density-based GMDBSCAN-UR clustering algorithm. We propose a grid-based cluster technique to reduce the time complexity. Grid-based technique divides the data space into cells. A number of well scattered points in each cell in the grid are chosen. These scattered points must capture the shape and extent of the dataset as all. Thus, our work in adopts a middle ground between the centroid-based and the all-point extremes. Next we treat all data in the same cell as an object, and all the operations of clustering are done on this cell. We make local clustering in each cell and merge between the resulted clusters. We use local MinPts for every cell in the grid to overcome the problem of undetermined clusters in multi-density datasets in clustering with DBSCAN clustering algorithm case. This will enhance the time complexity. Next step is labelling the not chosen points to the resulted clusters. Finally, we make post processing and noise elimination.

Keywords: Clustering, DBSCAN, Multi-density DBSCAN, Grid-based MDBSCAN, MDBSCAN-Using Representatives.

I. INTRODUCTION

Clustering is the process of grouping the data into classes or clusters, so that objects within a cluster have high similarity in comparison to one another but are very dissimilar to objects in other clusters. Dissimilarities are assessed based on the attribute values describing the objects. Often, distance measures are used [1]. The field of clustering has undergone major revolution over the last few decades; it has its roots in many areas, including data mining, statistics, biology, and machine learning. Clustering is characterized by advances in approximation and randomized algorithms, novel formulations of the clustering problem, algorithms for clustering massively large data sets, algorithms for clustering data streams, and dimension reduction techniques [2].

We study the requirements of clustering methods for large amounts of data and explain how to compute dissimilarities between objects represented by various attribute or variable types. Several studies examine a lot of clustering techniques, organized into the following categories: partitioning methods, hierarchical methods, density-based methods, grid-based methods, model-based methods, methods for high-dimensional data (such as frequent pattern–based methods), and constraint-based clustering [1]. Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge which can be used for applications ranging from market analysis, fraud detection, and customer retention, to production control and science exploration. Data mining can be viewed as a result of the natural evolution of information technology in a lot of functionalities such as data collection and database creation, data and advanced data analysis (involving data warehousing and data mining) [3].

Data clustering, also called cluster analysis, segmentation analysis, taxonomy analysis, or unsupervised classification, is a method of creating groups of objects, or clusters, in such a way that objects in one cluster are very similar and objects in different clusters are quite distinct. Data clustering is often confused with classification, in which objects are assigned to predefined classes. There are many algorithms used for clustering such that: hierarchical clustering techniques, fuzzy clustering algorithms, center-based clustering algorithms, search-based clustering algorithms, graph-based clustering algorithm, grid-based clustering algorithms, density-based clustering algorithms, model-based clustering algorithms, subspace clustering [1].

There are many algorithms that deal with the problem of clustering large number of objects. The different algorithms can be classified regarding different aspects. These methods can be categorized into partitioning methods [4, 5, 6], hierarchical methods [4, 7, 8], density based methods [9, 10, 11], grid based methods [12, 13, 14], and model based methods [15, 16].

Here in this research, we concentrate around the topic of DBSCAN algorithm, (Density-Based Spatial Clustering of Applications with Noise), and enhance it at all, time and space complexity, support multi-density grid based clustering in effective and efficient way.

DBSCAN checks the Eps-neighborhood of each point in database. If Eps- neighborhood of a point p contains more than MinPts, a new cluster with p as a core object is created. It then iteratively collects directly density-reachable objects from these core objects, which may involve the merge of a few density-reachable clusters. The process terminates when no new point can be added to any cluster [17]. The conventional DBSCAN and its improved algorithms presented in papers [18, 19, 20, 21] can only process the numerical data. They are incapable of processing data with categorical attributes.

Usually, the densities of dataset used in cluster analyses are different, however, until now there is no a very effective algorithm to get the accurate density of the dataset with multi-density. DBSCAN [18], density-based clustering not only availably avoids noises but also effectually clusters various datasets, whereas; for the multi-density dataset, DBSCAN is not a good algorithm for which the runtime complexity is high [1]. In order to reduce the time complexity, the academia has presented a grid-based cluster technique [22, 23], which divides the data space into disjunctive grid. The data in the same grid can be treated as a unitary object, and all the operations of clustering are on the grid [22].

This paper is principally concerned with the theoretical and experimental study of a set of multi-density clustering algorithms. And then make improvements on these clustering algorithms results in both quality and time.

The novelty of the proposed approach in this paper is that the we developed a new clustering algorithm named "GMDBSCAN-UR", Grid-based Multi-density DBSCAN Using Representative, by using sp-tree for clustering complicated and complex shaped datasets in a fast and accurate fashion based on grid and uses representative points that represent the dataset which leads to give the clustering result in an early time compared with using all points in the datasets which leads to a time consuming. Then the remaining points are labeled to the clusters based on that each non representative point to which cluster is the corresponding nearest representative point belongs. Experimental results are shown in this thesis to demonstrate the effectiveness of the proposed algorithms. We compared our proposed algorithm results with other famous related algorithms results. And we present that our new proposed algorithm is the best one in both quality and time.

II. RELATED WORKS

Clustering has been studied extensively for more than 40 years and across many disciplines due to its broad applications. Most books on pattern classification and machine learning contain chapters on cluster analysis or unsupervised learning. Several textbooks are dedicated to the methods of cluster analysis.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density based clustering algorithm. The algorithm grows regions with sufficiently high density into clusters and discovers clusters of arbitrary shape in spatial databases with noise. It defines a cluster as a maximal set of density-connected points [1].

The key idea of density-based clustering is that for each object of a cluster the neighborhood of a given radius (Eps) has to contain at least a minimum number of objects (MinPts), i.e. the cardinality of the neighborhood has to exceed a threshold [24]. DBSCAN [25] algorithm has many problems. It needs to know two parameters: Eps and MinPts and the value of parameter Eps is important for DBSCAN algorithm, but the calculation of Eps is a time-consuming. It must draw a sorted k-dist graph for dataset and user determines the first valley as the threshold Eps in the graphical representation.

Although DBSCAN can cluster objects given input parameters such as □ and MinPts, it still leaves the user with the responsibility of selecting parameter values that will lead to the discovery of acceptable clusters. Actually, this is a problem associated with many other clustering algorithms. Such parameter settings are usually empirically set and difficult to determine, especially for real-world, high-dimensional datasets. Most algorithms are very sensitive to such parameter values: slightly different settings may lead to very different clustering of the data. Moreover, high-dimensional real datasets often have very skewed distributions, such that their intrinsic clustering structure may not be characterized by global density parameters. To help overcome this difficulty, a cluster analysis method called OPTICS was proposed [10].

DENCLUE (DENsity-based CLUstEring) [1] is a clustering method based on a set of density distribution functions. The method is built on the following ideas: (1) the influence of each data point can be formally modeled using a mathematical function, called an influence function, which describes the impact of a data point within its neighborhood; (2) the overall density of the data space can be modeled analytically as the sum of the influence function applied to all data points; and (3) clusters can then be determined mathematically by identifying density attractors, where density attractors are local maxima of the overall density function.

MDBSCAN is a new method incorporates pairwise constraints (must-link) [26] in order to calculate parameters effectively and automatically which was used to deal with multi-density datasets. It makes use of some must-link constraints to calculate some parameters Eps in different density distributions; in latter step, it selects the best parameter Eps that reflects the current density distribution effectively for each density distribution by using a certain outlier detection algorithm; finally, MDBSCAN works on the multi-density data set with the calculated Eps [17].

Due to DBSCAN algorithm cannot choose parameter according to distributing of dataset. It simply uses a global MinPts parameter, so that the clustering result of multi-density database is inaccurate. In addition, when it is used to cluster large databases, it will cost too much time. For these problems, GMDBSCAN algorithm [25], based on spatial index and grid technique, is proposed.

The Problems of GMDBSCAN is a time consuming to perform well on large datasets, and sometimes it gives the output after a long time.

CURE was proposed as a hierarchical clustering algorithm that adopts a middle ground between the centroid-based and the all-point extremes [8]. CURE algorithm is more robust to outliers, and identifies clusters having non-spherical shapes and wide variances in size. It achieves this by representing each cluster by a certain fixed number of points that are generated by selecting well scattered points from the cluster, the scattered points capture the shape and extent of the cluster. Then it is shrinking them toward the center of the cluster by a specified fraction. Having more than one representative point per cluster allows CURE to adjust well to the geometry of non-spherical shapes and the shrinking helps to dampen the effects of outliers.

The grid-based clustering approach uses a multiresolution grid data structure. It quantizes the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed.

In general, a grid-based clustering algorithm consists of the following five basic steps:

a. Partitioning the data space into a finite number of cells (or creating grid structure).
b. Estimating the cell density for each cell,
c. Sorting the cells according to their densities,
d. Identifying cluster centers,
e. Traversal of neighbor cells.

The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects, yet dependent on only the number of cells in each dimension in the quantized space. It significantly reduces the computational complexity. Some typical examples of the grid-based approach include STING, which explores statistical information stored in the grid cells; WaveCluster, which clusters objects using a wavelet transform method; and CLIQUE, which represents a grid-and density-based approach for clustering in high-dimensional data space. OptiGrid, GRIDCLUS, GDILC, WaveCluster are also examples of grid-based clustering [27].

III. PROPOSED ALGORITHM "GMDBSCAN-UR"

The purpose of this research is to discover clusters with arbitrary shape, to regard clusters as dense regions of objects in the data space that are separated by regions of low density representing noise. In addition, the study is interested in algorithms that take into account the density to cluster the various real and artificial datasets. Our work in this research performs the density-based clustering in many stages as you can see in the following sections.

A. Adopted Idea:

What exactly happens in practice is as follows: to begin with, the first stage is to input a multi-density dataset. We want to reduce the time complexity. In order to achieve this purpose, we propose a grid-based cluster technique, [22, 23] which divides the data space into cells. A number of well scattered points in each cell in the grid are chosen. These scattered points capture the shape and extent of the dataset as all. These scattered points after shrinking are used as representatives of its cell. The chosen scattered points are next shrunk towards the centroid of the cell by a fraction alpha. The cells with the closest pair of representative points are the cells that are merged at each step of our work. The scattered points approach employed by our work alleviates the shortcomings of both the all-points as well as the centroid-based approaches [28]. Thus, our work in this research adopts a middle ground between the centroid-based and the all-point extremes. Next we treat all data in the same cell as an object, and all the operations of clustering are done on the cell. This research deals with two approaches of making clustering data set with multi-densities; the two choices give the same result with flexible options.

The first option is dealing with a specific cell with its local density so that; it is possible to vary the parameter Eps from cell to cell in the dataset and make the parameter MinPts to be constant. The second option is to make the parameter Eps constant over all cells and vary the parameter MinPts from cell to cell. Parameters choice depends on the local cell density. Next, make local clustering in each cell and merge between the resulted clusters. The next step is labeling the points, not chosen in the representative points, to the resulted clusters. After that, the post processing is become needed in such a way to accurate the results. Finally and the necessary step in all clustering algorithms is to eliminate noise and outliers.

B. GMDBSCAN-UR Steps:

Our proposed clustering algorithm, GMDBSCAN-UR, consists of eight steps, these are as follows:

a) Datasets input and Data standardization.
b) Dividing dataset into smaller cells.
c) Chosen representative points in each cell.
d) Selecting Parameters of MinPts and Eps.
e) Bitmap Forming.
f) Local-Clustering and Merging the Similar Sub-clusters using DBSCAN algorithm.
g) Labeling and Post Processing.
h) Noise Elimination

Here we explain our new multi-density clustering algorithm based on grid and using representative points, GMDBSCAN-UR.

a. Datasets Input and Data Standardization:

Data standardization [29] makes data dimensionless. It is useful for defining standard indices. After standardization, all knowledge of the location and scale of the original data may be lost. It is necessary to standardize variables in cases where the dissimilarity measure, such as the Euclidean distance, is sensitive to the differences in the magnitudes or scales of the input variables [30]. The approaches of standardization of variables are essentially of two types: global standardization and within-cluster standardization. Global standardization standardizes the variables across all elements in the data set. Within-cluster standardization refers to the standardization that occurs within clusters on each variable. Some forms of standardization can be used in global standardization and within-cluster standardization as well, but some forms of standardization can be used in global standardization only [27]. The z-score is a well known form of standardization used for transforming normal variants to standard score form. Given a set of raw data D, the z-score standardization formula is defined as

$$x_{ij} = z_1\left(x_{ij}^*\right) = \frac{x_{ij}^* - x_{ij}^*}{\sigma_j^*} \qquad 1$$

Where $\bar{x}_{ij}^*$ and σ_j^* are the sample mean and standard deviation of the jth attribute, respectively. The transformed variable will have a mean of 0 and a variance of 1. The location and scale information of the original variable has been lost. This transformation is also presented in [31].

So, the first step in our proposed algorithm, GMDBSCAN-UR, is to input the dataset which is in a multi-density format like for example real dataset "adult" and artificial data set "chameleon", we name it like that because it has been used to evaluate chameleon algorithm [29]. And then make the data standardization step. Figure 1 below is a multi-density dataset; it has clusters with different densities.

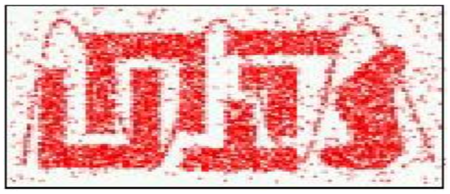
Figure 1. Multi-density dataset.

b. *Dividing Dataset into Smaller Cells:*

Partitioning divides the data space into smaller cells. So the cells numbers of points are not equal. Figure 2 shows a multi-density example which is divided to cells as in Figure 3 which shows that the top most left cell's number of points is not equal to top most right cell's number of points. So the second step is dividing the data space into cells in order to make local clustering in each cell. The number of cells per dimension is calculated in SPTree.construct() method. The SP-Tree is created only on these dense cells where CD >= I.
where: I denotes the density threshold; it is an integer value and, CD is the cell density.

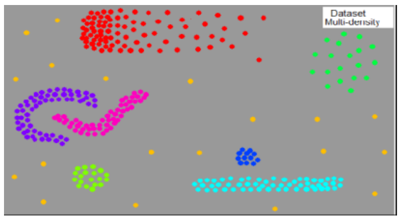

Figure 2. Multi-density example.

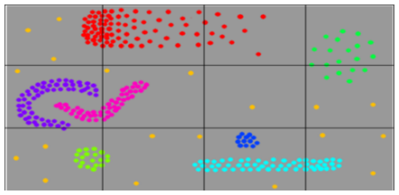

Figure 3. Dividing the dataset to smaller cells.

Then each point is then assigned to a cell by the SPTree.insert() method. Cells i.e. the SPTree.leaves, have a collection of points.

c. *Chosen Representative Points:*

A number of well scattered points in each cell are chosen. The scattered points in the cell must capture the shape and extent of that cell as shown in Figure 4. The black points in Figure 4 below are the representative points, it is clear that the number of representative points is smaller than the number of points in the cell. And these representative points are well scattered over the original dataset points. This step leads to significant improvements in execution times in our new proposed GMDBSCAN-UR algorithm.

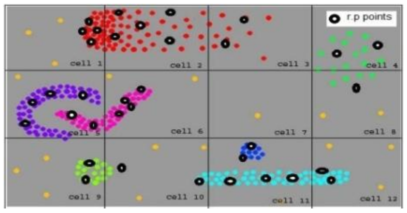

Figure 4. Taking a well scattered representative points in each cell.

So we visit all cells (leaves) in the tree and choose a percentage number of points, say half, to be the representative points in the cell. All the representative points in all cells are the representative points of the dataset. Figure 5 below shows some dataset along with its representative points. It is clear that the chosen representative points Figure 5 (b) are actually represents the original dataset Figure 5 (a). Good representing the original dataset is very important issue in getting good final clustering results.

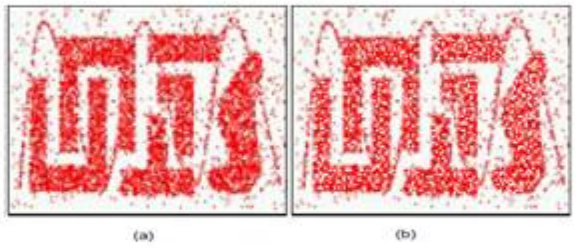

Figure 5. Dataset along with its representative points.

d. *Selecting MinPts and Eps Parameters:*

In each cell, one approach is used in selecting the MinPts and Eps. Either selects the MinPts for each cell individually and let the Eps to be constant for all cells or select the Eps for each cell individually and let the MinPts to be constant for all cells.

a) ***Firstly:*** when we use the same Eps with varying MinPts, then we have the following:

We apply the idea on the cells as shown below in Figure 6 using same MinPts in all cells to merge but in different Eps from cell to cell; i.e. the MinPts is 4 at all cells but, at the most left grid the Eps is the smallest because this cell is the most dense; at the middle cell the Eps is wider because this cell is less dense, at the right cell the Eps is the widest because it is the lowest density.

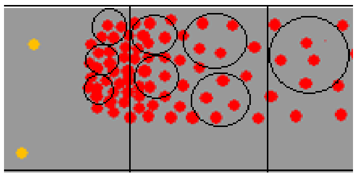

Figure 6. Using same MinPts with varying Eps.

b) ***Secondly:*** we apply the idea using the same Eps with varying MinPts, then we have the following: we apply the idea on two cells as shown below in Figure 7; using same Eps in all cells to merge but in different number of MinPts from cell to cell, i.e. at the left cell the MinPts is 4; the right choosing the cell MinPts to be 2 is enough.

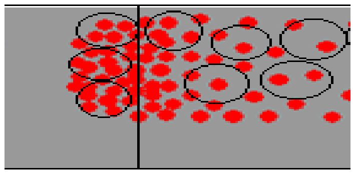

Figure 7. Using same Eps with varying MinPts.

e. *Bitmap Forming:*

Calculate the distance of two data which exists in the same or adjacent cells. Calculate the distances of each two data and compare with Eps then store the information in the bitmap. If the distance is less than or equal to Eps, it means the data are in each other's neighborhood [27].

Cell Density (CD) is defined as amount of data in a cell. We take CD as local-MinPts approximation of the cell region. If the Cell Volume, denoted by, VCell is not equal to the data point's neighborhood volume, VEps, we set a factor to correct it, factor = VEps / VCell. The relationship of CD and MinPts is:

Factor = MinPts / CD = VEps / VCell 2

MinPts = factor * CD 3

This step makes all necessary needed statistics which will be used in the next later steps.

f. Local-Clustering and Merging the Similar Sub-clusters using DBSCAN Algorithm

In this step we apply the original DBSCAN method locally in each cell using the computed MinPts and Eps parameters. Our work in this study mainly gets the idea of locally clustering, identifying a local-MinPts for each cell in the dataset. For each cell, processing clustering with their local-MinPts to form a number of distributed local clusters.

The step is divided into consecutive steps to make local clustering and merging the similar sub clusters as shown.

a) First is to select the cell whose density is maximal and has not been clustered. During that time we deal with boundary. Dense cells refer to those cells whose cell density, CD is greater than or equal to some predefined threshold.

b) Then sparse cells which close to dense cells, but their cell densities are less than the pre-specified threshold. Data in sparse cell may be noise or border, it needs further study. Isolated cells refer to those whose cell density is less than threshold and not close to some dense cell. All data in isolated cells could be regarded as noise and isolated data. In DBSCAN, if the border object is in the scope of Eps neighborhood of different core objects, it is classified into the cluster to sort firstly. Here in our GMDBSCAN-UR algorithm, we set such object to the cluster whose core object is the nearest to this object. Second, we compute MinPts for each data in cell which its equation given by:

MinPts = factor * CD 4

c) Then Cluster with original DBSCAN algorithm and for each unvisited point, P, in dataset, D, mark P as visited and compute the neighbors of the point P, then compare this number with the MinPts. If neighbors are less than MinPts then label P as a noise, otherwise label it as a core point and so on. Then expand the current cluster.

d) If data belongs to another sub-cluster, then merge the two clusters, and if not, assign it to the cluster whose has the nearest representative point from this point and tag the data as a new cluster.

g. Labeling and Post Processing:

Since the input to GMDBSCAN-UR's clustering algorithm is a set of well scattered chosen representative points from the original large dataset, the final k clusters involve only a subset of the entire set of points. In GMDBSCAN-UR, the algorithm for assigning the appropriate cluster labels to the remaining data points employs the selected representative points for each of the final k clusters. Each data point is assigned to the cluster containing the representative point closest to it. Note that approximating every cluster with multiple points instead a single centroid enables GMDBSCAN-UR correctly distribute the data points when clusters are non-spherical or non-uniform. After that, we check the density for the resulted k cluster. If there are two clusters with nearly same density and very close to each others, they can be merged to a single cluster as a post processing step. The remerging step goes out with very accurate final results. Thus, the remerging step applied on the resulted clusters is very necessary step.

Labeling and Post Processing step labeling all points in the dataset_Remainder data set, which are not entered in the clustering process. For every point in the dataset_Remainder, we search for the nearest point from the dataset_Rep dataset, which is clustered and becomes point, belongs to some cluster. Once find the nearest point from the dataset_Rep to the point from dataset_ Remainder, we can label this non-clustered point to the cluster contains the nearest representative point. Now, to this end we have all points in the dataset become clustered.

h. Noise Elimination:

Any data set almost always contains outliers. These do not belong to any of the clusters. That is, the neighborhoods of outliers are generally sparse compared to points in clusters, and the distance of an outlier to the nearest cluster is comparatively higher than the distances among points in points in the clusters themselves. Every clustering method needs mechanisms to eliminate outliers. In GMDBSCAN-UR, outliers due to their larger distances from other points, tend to merge with other points less and typically grow at a much slower rate than actual clusters. Thus the clusters which are growing very slowly are identified and eliminated as outliers. Also, since the number of points in a collection of outliers is typically much less than the number in a cluster and that outliers form very small clusters, we can easily identify such small groups and eliminate them. Consequently, the final step, the outlier elimination, is necessary step for good clustering.

C. GMDBSCAN-UR Algorithm Properties:

a. Taking into account both the inter-connectivity as well as the closeness of the clusters.
b. Considering the internal characteristics of the clusters themselves.
c. Operates successfully on data sets with various shapes.
d. Dividing to Grid:
e. Allows scaling to large datasets.
f. Significantly, it reduces the computational complexity.
g. Does not depend on user-supplied model.
h. GMDBSCAN-UR clustering algorithm is very sensitive to two parameters, MinPts and Eps.
i. GMDBSCAN-UR clustering algorithm divides its work into three separate steps as follows:
   - ***a)*** ***First****:* Make the main clustering after dividing the data space to cells and choose the representative point to enter the clustering process with the DBSCAN algorithm.
   - ***b)*** ***Second****:* After getting the clusters and for the result to be more accurate, we need to perform such as a post processing step to get more accurate results of the clusters, that the result we get may contain more small size clusters due to over clustering which is not a good choice in clustering. So, here in GMDBSCAN-UR we run a remerging method to get more accurate result. This remerging method is not a time consuming and we can run such a method comfortably without worrying about the time to increase in the clustering algorithm.
   - ***c)*** ***Third****:* Up to this point we have a set of points that do not enter the clustering process and do not belong to

any of the resulting clusters. Here the role of the labeling step is to get every point that is not in the chosen representative set of points, i.e. not clustered points, and search for the nearest point to it from the representative points, and then assign the non-clustered point to the cluster that the representative clustered point belongs to. To this end, we have all data points are clusters and each belongs to the true cluster.

IV. EXPERIMENTAL RESULTS

In this section we will show a sufficient number of results with various types of datasets with various numbers of points also, and with these inputs we will compare between four algorithms namely: DBSCAN, MDBSCAN, GMDBSCAN and, our proposed algorithm, GMDBSCANUR for clustering 2 features (age versus fnlwgt (final weight) - subsets of UCI "adult" dataset), and another artificial dataset we name it chameleon because it used in evaluating Chameleon clustering algorithm. The following subsections contain figures and tables which show the corresponding times and the number of clusters for each comparable algorithm.

A. Adult Dataset:

This dataset comes from the UCI repository of machine learning databases. It is a multivariate dataset as we want to show the effectiveness of our proposed GMDBSCAN-UR algorithm. Adult dataset has 48842 categorical and integer instances. It has 14 attributes. Figure 8 below shows adult dataset. In our experiments we choose two numerical, integers, attributes. So, the dataset contains three clusters each has a different density.

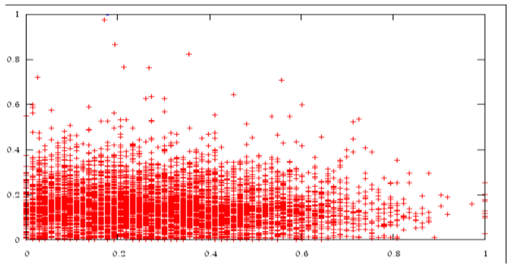

Figure 8. Adult dataset.

B. Chameleon Dataset:

Dataset is an artificial dataset; we use it to evaluate our proposed algorithm. It has 8000 data points. We choose two attributes of double values. Chameleon dataset has a nested not simple shape dataset and it consists of six multi-densities clusters. Each cluster has arbitrary shape and many noises. Figure 9 shows chameleon dataset.

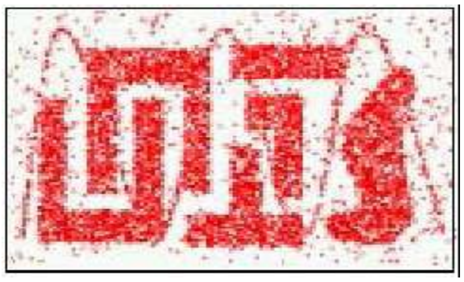

Figure 9. Chameleon dataset.

C. DBSCAN Results:

In this section we are going to explain DBSCAN clustering algorithm results. DBSCAN fails in clustering the multi-densities datasets like adult. It fails in clustering various numbers of points; that it cannot discover all clusters in the dataset. DBSCAN can only discover one or two clusters and fail to discover the others. In clustering 250 points, it can discover only one cluster, while the others disappeared because DBSCAN was not able to appear them when we cluster with all points, 8000 points, in the dataset. Figures 10, shows the results of clustering all the 8000 point from chameleon dataset with DBSCAN clustering algorithm. Although we use all the dataset's data points, DBSCAN only discover one cluster.

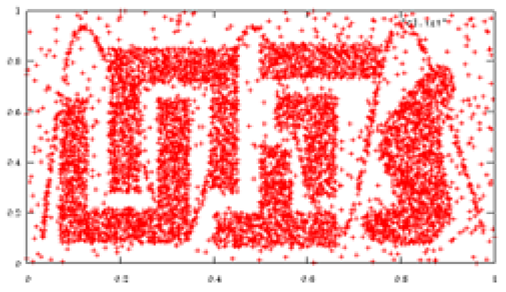

Figure 10. Chameleon dataset clustering result using DBSCAN algorithm.

DBSCAN merges between different clusters whereas it is impossible to merge between them. The previous wrong merges between clusters result with a wrong final results. So, DBSCAN is a very bad clustering algorithm with a multi-density datasets.

D. MDBSCAN Results:

Here in this section, we will offer the results of the second comparison algorithm. MDBSCAN gives results better than DBSCAN results in clustering the multi-densities datasets like adult. MDBSCAN was not able to discover all clusters in the dataset correctly. In clustering 250 points, it can discover only two cluster, while in clustering 500 points, it can discover only one cluster. The most important point to talk about is the result of clustering the dataset at all, 8000 points. Figure 11 are the results of clustering all the 8000 point from chameleon dataset with MDBSCAN clustering algorithm. MDBSCAN algorithm makes more not needed and wrong splits and merges.

Clustering with MDBSCAN is a very time consuming process. For this reason, we stopped clustering more points beyond the 1000 points. So, MDBSCAN is a very time consuming clustering algorithm and not a good choice when clustering a large datasets.

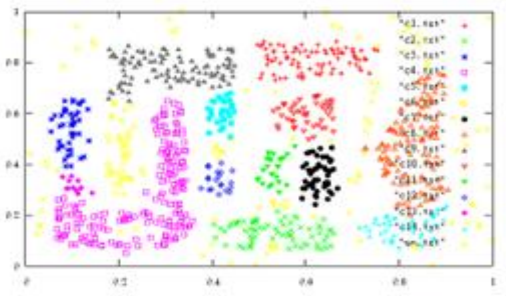

Figure 11. Chameleon dataset clustering result using MDBSCAN algorithm.

E. GMDBSCAN Results:

This is an improved version of DBSCAN algorithm. It is a multi-density clustering algorithm. We can see that GMDBSCAN algorithm gives good results but it takes a bit more time which makes us to search for a better one in clustering time. Figure 12 shows GMDBSCAN algorithm clustering results.

GMDBSCAN algorithm results is better than MDBSCAN algorithm's results in both quality and time.

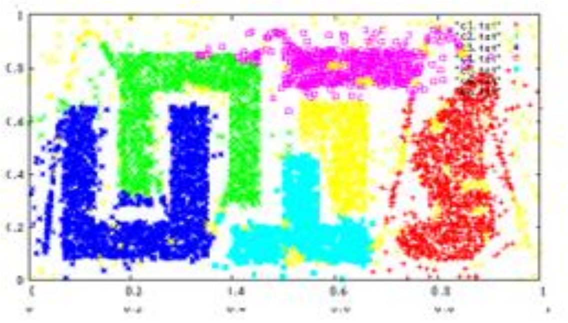

Figure 12. Chameleon dataset clustering result using GMDBSCAN algorithm.

F. GMDBSCAN-UR Results:

GMDBSCAN-UR clustering algorithm solves all problems we faced in all previous algorithms as we can see from the results in this section. Figure 13 illustrates the clustering results from chameleon datasets with GMDBSCAN-UR clustering algorithm. Here, we will talk in more depth about the clustering process with our new proposed GMDBSCAN-UR clustering algorithm. In this section, we evaluate the performance of GMDBSCAN-UR, and compare it with previous density-based clustering algorithms. All problems we faced in all previous density-based clustering algorithms are solved with GMDBSCAN-UR algorithm. GMDBSCAN-UR gets better results than GMDBSCAN results. GMDBSCAN-UR can recognize noise and outliers from the datasets. Chameleon dataset has 8000 data points of six clusters. Each cluster has arbitrary shape and many noises. Our proposed clustering algorithm, GMDBSCAN-UR, succeeded in clustering adult and chameleon multi-density datasets and gives good quality results with a short times. The performance of GMDBSCAN-UR algorithm is superior to GMDBSCAN, MDBSCAN and DBSCAN algorithms as the volume of data increases. In GMDBSCAN-UR algorithm, if the data points of dataset increased, the runtime complexity increases linearly as the volume of data increases. At the same time, the improvements in time compared with the previously mentioned density-based algorithms are increased very fast.

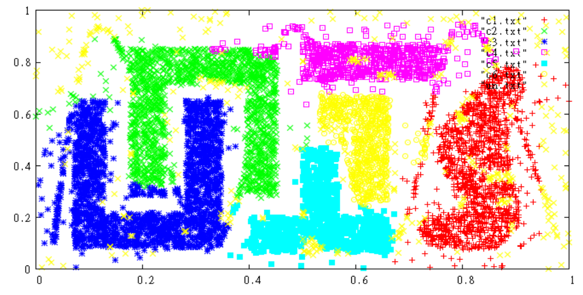

Figure 13. Chameleon dataset clustering result using GMDBSCAN-UR algorithm.

From the result in Figure 13 we see that GMDBSCAN-UR algorithm gets accurate clusters, and also recognizes noises with sparse distribution. GMDBSCAN-UR algorithm does better in case of a lot of noises with more intensive distribution existence.

GMDBSCAN-UR gets this result in nearly 83 seconds in clustering adult dataset and 30 seconds in clustering chameleon dataset, i.e. this time is better than the other density-based algorithms' clustering times as we see in the following tables.

Here we will illustrate, in more deeply, the comparison between the four clustering algorithms in terms of both the quality of the resulting clusters and the time each takes to produce there results. Table I first row represents the density-based clustering algorithms GMDBSCAN-UR, GMDBSCAN, MDBSCAN and DBSCAN. Table 1 is the resulted times of clustering "adult" and "chameleon" datasets with the above mentioned four clustering algorithm. Table's first column represents the number of points used to make the clustering process. These numbers are 250,500,1000,2000,4000 and 8000 points, the table's first column. The table second and fourth columns are the multi-density dataset's, adult and chameleon, clustering times in milliseconds. While third and fifth columns are the corresponding resulted numbers of clusters.

Table I. Adult dataset comparative clustering results summary using the four algorithms.

No. Points	**GMDBSCAN-UR**	**GMDBSCAN**	**MDBSCAN**	**DBSCAN**
	Time (ms)	**Time**	**Time**	**Time**
250	140	218	499	47
500	405	826	1623	140
1000	1436	3027	5631	796
2000	5696	13579	20891	5522
4000	14978	29329	502486	41341
8000	83975	186506	very long time	390855

By looking at the results in more depth at the following table, we will note how the differences in times between the four algorithms are clear for a various numbers of points used. The GMDBSCAN takes more than twice the time GMDBSCAN-UR algorithm takes. And DBSCAN nearly takes the time equal to five times the time GMDBSCAN-UR takes to cluster 8000 points in adult dataset. Our experiments results show how wonderful the output of our new proposed algorithm, GMDBSCAN-UR, with respect to other related algorithms in both the quality and in time. It is really strong, active and fast in finding clusters effectively.

Note that clustering small datasets made all the four clustering algorithms take nearly the same times, and this is not our concern. We are interested in clustering large datasets which contains a number of thousands data points to show the efficiency and the quality of the clustering algorithm.

For Chameleon dataset and by clustering few number of points, say 250 point, we see that DBSCAN algorithm is the smallest numbers of times and the MDBSCAN is the worst one and in some cases when the dataset is large that contains a large number of points, MDBSCAN takes a very long time to cluster it. In a large and complex datasets, using a large number of data points, say 4000 or 8000 points, the

GMDBSCAN-UR is the best one in results for both quality and the times it takes to produce the results, see Tables II.

Table II. Chameleon dataset comparative clustering results summary using the four algorithms.

No. Points	GMDBSCAN-UR	GMDBSCAN	MDBSCAN	DBSCAN
	Time (ms)	Time	Time	Time
250	94	125	1763	47
500	203	203	64608	141
1000	796	967	426040	453
2000	2809	6008	Failed	2527
4000	10751	19579	Failed	19282
8000	30767	59470	Failed	159115

Then, to support our idea in proving that the GMDBSCAN-UR is the best one, we use two datasets, chameleon and adult as we see from the above Tables. GMDBSCAN-UR clustering algorithm runs in three separate steps. Each step takes its separate time. For example, the three steps times for clustering chameleon dataset with GMDBSCAN-UR algorithm is as follows for 8000 points:

a. The time after first step, main clustering step is: 17044 ms.
b. The time after the second step, remerge step: 19449 ms.
c. The time after the third step, labeling step : 30767 ms.

V. CONCLUSIONS

In this study, we introduce a new multi-density clustering algorithm based on grid and uses representative points that take the general shape of the data in the dataset. We perform an experimental evaluation to the performance of GMDBSCAN-UR using real data .The results of our experiments show that GMDBSCAN-UR is effective and efficient. In this study, in addition to handling data sets which are high dimensional, we also use representative points technique to work with reduced number of points in the dataset which result in a high saving in time. We investigated using representative points not all data set points for improving the performance of our algorithm. We developed a novel effective clustering algorithm which improved the performance of the DBSCAN algorithm.

The proposed clustering algorithm uses SP-tree, and divides its work into three main steps which are main clustering after getting the representative points, remerging clusters to get a more accurate result and last labeling the remainder data points which are not entered in the clustering process to the true clusters that they belong to. The GMDBSCAN-UR algorithm that we presented is specific to clustering more complex and with large number of points data sets. The proposed clustering algorithms have a great saving in running time and giving amazing results. Experimental results are shown in this thesis to demonstrate the effectiveness of the proposed algorithm. We illustrated the time complexity and the performance of classifying complex data sets. We proved that the proposed algorithms can classify complex data sets more accurately than other previous algorithms.

VI. REFERENCES

[1]. J. Han and M. Kamber, "Data Mining: Concepts and Techniques," Morgan Kaufmann Publishers, 2006.

[2]. Nina Mishra "Clustering Algorithms" available at http://theory.stanford.edu/~nmishra/cs369C-2005.html , last visit 22 Aug 2011

[3]. J. Han andM. Kamber. "Data Mining: Concepts and Techniques," Morgan Kaufmann Publishers, 2001.

[4]. L. Kaufman and P. Rousseeuw, "Finding groups in Data: an Introduction to cluster," John Wiley & Sons, 1990.

[5]. J. Han, M. Kamber, and A. K. H. Tung, "Spatial Clustering Methods in data mining," A Survey, Geographic Data Mining and Knowledge Discovery, 2001.

[6]. P. Bradly, U. Fayyad, and C. Reina, "Scaling clustering algorithms to large databases," In proc. 1998 Int. Conf. Knoweldge Discovery and Data mining, 1998.

[7]. T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: an efficient data clustering method for very large databases," In Proc. 1996 ACMSIGMOD Int. Conf. Management of data (SIGMOD'96), 1996.

[8]. S. Guha, R. Rastogi, and K. Shim, "Cure: An efficient clustering algorithm for large databases," In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98), 1998.

[9]. M. Ester, H. P. Kriegel, J. sander, and X. Xu, "A density based algorithm for discovering clusters in large spatial databases," In Proc. 1996 Inc. Conf. Knowledge discovery and Data mining (KDD'96).

[10]. M. Ankerst, M. Breunig, H.P. kriegel, and J. Sander, "OPTICS: Ordering points to identify the clustering structure," In Proc. 1999 ACM-SIGMOD Int. Conf. Management of data (SIGMOD'96), 1999.

[11]. A. Hinneburg and D. A. Keim, "An efficient approach to clustering in large multimedia databases with noise," In Proc. 1998 Int. Conf. . Knowledge discovery and Data mining (KDD'98), 1998.

[12]. W. Wang, J. Yang, and R. Muntz, "STING: A statistical information grid approach to spatial data mining," In Proc. 1997 Int. Conf. Very Large Data Bases (VLDB'97), 1997.

[13]. G. Sheikholeslami, S. Chatterjee, and A. Zhang, "Wave Cluster : A multi- resolution clustering approach for very large spatial databases," In Proc. 1997 Int. Conf. Very Large Data Bases (VLDB'97), 1998.

[14]. R. Agrawal, J. Gehrke and D. Gunopulos, P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining application," In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98), 1998.

[15]. J. W. Shavlik and T.G. Dietterich, "Reading in machine learning," 1990.

[16]. T. Kohonen, "Self organized formation of topologically correct feature maps," Biological Cybernetics, 1982.

[17]. T. Huang ,Y. Yu, K Li and W. Zeng, "Reckon the Parameter of DBSCAN for Multi-density Data Sets with Constraints," Dept. of Computer Science, School of Mathematics & Computer Science Fujian Normal University Fuzhou, China; 2009.

[18]. Martin Ester and Hans-Peter Kriegel, "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc of 2nd Int. Conf. on KDD'96, 1996: 226-231.

[19]. Zhou S, Zhou A and et al, "A Fast Density-Based Clustering Algorithm," [J]. Journal of Computer Research and Development, 2003, 37(11):1287-1292.

[20]. Zhou S, Fan Y and Zhou A. SDBSCAN, "A Sampling- Based DBSCAN Algorithm for Large-Scale Spatial Databases,"

Journal of Chinese Computer Systems[J], 2000, 21(12): 1270-1274.

[21]. Zhou A, Zhou S, Cao J and et al. "Approaches for scaling DBSCAN algorithm to large spatial database," [J].Journal of computer science and technology,2000,15(06): 509-526.

[22]. Zeng Donghai, "The Study of Clustering Algorithm Based on Grid-Density and Spatial Partition Tree," XiaMen University, PRC, 2006.

[23]. A. H. Pilevar. M. Sukumar, "GCHL: a grid-c1ustering algorithm for high-dimensional Data Mining," SBIA 2004, LNAI 3171.

[24]. M. Ester and H. Kriegel. "A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc of 2nd Int. Conf. on KDD'96, 1996: 226-231.

[25]. C. Xiaoyun, M. Yufang, Z. Yan and W.Ping, "GMDBSCAN: Multi-Density DBSCAN Cluster Based on Grid," School of Information Science and Engineering, Lanzhou University Lanzhou 730000, PRC China.

[26]. K. Wagstaff, C. Cardie, S. Rogers and S. Schroedl, "Constrained K-Means clustering with background knowledge," Proc. of 18th Int. Conf. on Machine Learning (ICML-2001), Morgan Kaufmann Publishers, Jun. 2001, pp.577-584, doi: 10.1.1.20.7363.

[27]. Gan, G., Ma, C., and Wu, J.," Data Clustering: Theory, Algorithms, and Applications,"

[28]. S. Guha, R. Rastogi and K. Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," Stanford University.

[29]. A. H. Pilevar and M. Sukumar, "GCHL: a grid-c1ustering algorithm for high-dimensional Data Mining," SBIA 2004, LNAI 3171.

[30]. G. Milligan and M. Cooper (1988), "A study of standardization of variables in cluster analysis," Journal of Classification, 5:181–204.

[31]. A. Jain and R. Dubes (1988). Algorithms for Clustering Data. Englewood Cliffs, NJ: Prentice–Hall.