2006 IEEE International Conference on
Systems, Man, and Cybernetics
October 8-11, 2006, Taipei, Taiwan

GRIDBSCAN: GRId Density-Based Spatial Clustering of Applications with Noise

Ozge Uncu, *Member, IEEE*, William A. Gruver, *Fellow, IEEE*, Dilip B. Kotak, *Senior Memmber, IEEE*
Dorian Sabaz, *Member, IEEE*, Zafeer Alibhai *Member, IEEE*, and Colin Ng, *Member, IEEE*

***Abstract*—Clustering is one of the basic data mining tasks that can be used to extract hidden information from data in the absence of target classes. One of the most well-known density based clustering algorithms for processing spatial data is Density-Based Spatial Clustering of Application with Noise (DBSCAN) that uses learning parameters ε and *minPts* to define the density that will be sought in the data set while forming the clusters. The major drawbacks of the DBSCAN algorithm are its sensitivity to user input required to execute the algorithm, inability to recognize clusters with different densities, and computational complexity. In this study, we propose a three-level clustering method to address the second issue. The first level selects appropriate grids so that the density is homogeneous in each grid. The second stage merges cells with similar densities and identifies the most suitable values of ε and *minPts* in each grid that remain after merging. The third step of the proposed method executes the DBSCAN method with these identified parameters in the dataset. The proposed method is tested in three artificial benchmark data sets to demonstrate that the clusters are correctly identified.**

I. Introduction

Clustering involves the grouping of similar objects together. It is one of the fundamental data mining tasks that can serve as an independent data mining tool or a preprocessing step for other data mining tasks such as classification. Clustering is a versatile unsupervised learning method that can be used in several ways, e.g., outlier detection, data reduction and identification of natural data types and classes. In particular, some major clustering applications are marketing (finding groups of customers or consumers with similar behaviors in order to optimize marketing activities), World Wide Web (document clustering to optimize information retrieval and path traversal clustering to find users with similar click patterns to optimize the banners and caching), earth sciences (grouping earthquake epicenters to identify dangerous zones), and biology (groupings of DNA sequences).

Clustering methods can be categorized into partitioning, hierarchical, density-based, grid-based, and model-based methods [7]. Examples of partitioning methods are PAM [10], CLARA[10], and CLARANS[13]. These methods segment data into k groups where k is selected by the user. Examples of hierarchical methods are CURE [16] and CHAMELEON[9]) which build a tree like structure, called the Dendogram, to exhibit a cluster structure at every iteration. Density-based methods such as DBSCAN [5], DENCLUE [8], and OPTICS [2]) can determine clusters with arbitrary shapes by growing regions with sufficiently high density. CLIQUE [1], ENCLUS [3], and WaveCluster [15] are grid-based methods that segment the space into a finite number of cells in which all clustering operations are performed. Model-based clustering methods such as COBWEB [6] and Self Organizing Map (SOM) [11] optimize the fit between the given data and a mathematical model. Partitioning methods have relatively low complexity, however the number of clusters must be selected by the user. Hierarchical methods allow the user to see the sequence of clusters formed by dividing the clusters or aggregating the data to clusters, however, they have poor scalability relative to an increasing amount of data. Furthermore, a termination criterion is needed to decide the number of clusters. Although density-based methods can identify clusters with arbitrary shapes, most of these methods are sensitive to user input. Grid-based methods have low computational complexity. Their complexity generally depends heavily on the number of cells rather than the amount of data. Thus, grid-based methods are generally used to discover low dimensional clusters in high dimensional data [14].

DBSCAN [5] is the most popular density based clustering method to determine arbitrary shaped clusters in spatial databases containing noise. The user must provide parameter values ε and minPts to define the density of the clusters that will be sought in the database. Then, the concepts of density-reachability, density-connectivity, and density-connectivity are applied to grow clusters in the data space. The three major difficulties of the DBSCAN method are its sensitivity to user input, inability to find clusters with different densities, and computational complexity. Several studies have addressed these issues. In Xu, et al. [18] the first issue was attacked by proposing a method called DBCLASD that does not require user input. OPTICS [2] attacked the

This work received support through grants from an Industrial Partnership Grant from the Natural Sciences and Engineering Research Council of Canada, including contributions from the National Research Council of Canada and C. P. Loewen Enterprises Ltd., Steinbach, Manitoba, Canada.

O. Uncu is with Simon Fraser University, Burnaby, BC, V5A1S6, Canada (e-mail: ouncu@sfu.ca).

William A. Gruver is with Simon Fraser University, Burnaby, BC, V5A1S6, Canada. (e-mail: gruver@cs.sfu.ca).

Dilip. B. Kotak is with the National Research Council Canada, Vancouver, BC, V6T1W5, Canada (e-mail: Dilip.Kotak@nrc-cnrc.gc.ca).

Dorian Sabaz is with Intelligent Robotics Corp., North Vancouver, BC, V7H1K4, Canada (e-mail: dorian@iroboticscorp.com)

Colin Ng and Zafeer Alibhai is with Simon Fraser University, Burnaby, BC, V5A1S6, Canada. (e-mail: {cnge, zalibhai}@cs.sfu.ca)

second issue by providing a method to find clusters with different densities. "Rather than produce a data set explicitly, OPTICS computes an augmented cluster ordering for automatic and interactive clustering…It contains information that is equivalent to density-based clustering obtained from a wide range of parameter settings." [7]. Our proposed method also addresses the second issue. It differs from OPTICS by determining specific density parameter values for regions with different densities rather than using density-based clustering on all training data vectors with a wide range of parameter values.

II. GRIDBSCAN ALGORITHM

Because a major drawback of the DBSCAN algorithm is its inability to find clusters with different densities, the main goal of our proposed algorithm, GRIDBSCAN, is to provide a solution for this issue. Since the identification of the most suitable segments is provided in Uncu, et al. [17] we shall assume that most suitable set of cells have been identified in the first stage of the method. Then, the GRIDBSCAN method can be informally explained as follows: Let *NV* and *ND* be the number of the variables in the feature space and the number of data vectors in the data set, respectively. After selecting the most suitable segmentation points on each axis in the feature space, we let *NC* be the number of cells created by the selected points. Among *NC* cells, some attributes are identified for the cell that are worthwhile to process. The rationale for only treating those cells that are worthwhile to process is similar to that used in the ENCLUS method [3]. Thus, we define upper and lower bounds for the minimum data in cells to decide whether the cell is dense enough to be further process. As suggested in Cheng, et al. [3], a constant number of data is generally considered to be the minimum sample size for large sample procedures [4]. We define the lower and upper bound minimum data that will be denoted as *lb* and *ub,* respectively, as follows:

$$\begin{aligned} lb &= \min(0.01\times ND, 10) \\ ub &= \min(0.05\times ND, 30) \end{aligned} \tag{1}$$

where the constant ND is empirically chosen. e. Any cell that has less than *lb* data points is not worthwhile to process. The cells that contain data in the interval (*lb*,*ub*) will be treated as sparse cells that are worthwhile to be further processed. Otherwise, the cells will be considered to be data dense. The set of sparse cells *SC* and the set of dense cells *DC* are defined as

$$\begin{aligned} SC &= \{i \mid lb < ND_i < ub, \forall i = 1,..., NC\} \\ DC &= \{i \mid ND_i \geq ub, \forall i = 1,..., NC\} \end{aligned} \tag{2}$$

In our proposed method, the following attributes will be identified for sparse cells and dense cells: ND_i, ε_i, $minPts_i$, $indexD_i$, where ND_i and $indexD_i$ are the number and index of data points in the i^{th} cell, respectively, and ε_i and $minPts_i$ are the suitable parameters of DBSCAN assigned to the i^{th} cell, respectively, and i=1,…,|*DC*|+|*SC*|. Let the distance between the n^{th} and m^{th} data vectors z_m and z_n be denoted as $d(z_m,z_n)$. Then ε_i is determined by

$$\begin{aligned} \varepsilon_i &= \frac{\underset{m=1,...,ND_i}{median}(\varepsilon_{im})}{1.5}, \text{ where} \\ \varepsilon_{im} &= \underset{n=1,...,ND_i}{median}(d(z_m, z_n)) \end{aligned} \tag{3}$$

The median is used instead of the mean to increase the tolerance to outliers. The use of the median and the factor 1.5 can be justified by the application of the Small World Theory of Milgram [12] who reasons that any element in a population can be connected by using a constant number of nodes (elements) in the population. In most studies this constant has been selected to be 6. After determining ε_i, $minPts_i$ can be identified from

$$\begin{aligned} &\text{minPts}_i = \underset{m=1...ND_i}{median}\left|\text{NE}(z_m, \varepsilon_i)\right|, \text{ where} \\ &\text{NE}(z_m, \varepsilon_i) = \{z_n \mid d(z_m, z_n) \leq \varepsilon_i\}\ \forall m, n = 1,..., ND_i \\ &i \in \{t \mid ND_t \geq \min(10, 0.01\times ND), t = 1,...., NC\} \end{aligned} \tag{4}$$

The other cells are discarded because they do not have enough data vectors that have a significant role in identifying the density parameters of different clusters. This procedure is repeated until either all cells are processed or the total number of data in processed cells sums to *ND* (i.e., all remaining cells contain no data vectors).

It is assumed that the data dense cells are reliable in determining data and conversely. Thus, the attributes of the cells with indices in *SC* have been further adjusted by using the attributes of the immediate neighbor data-dense cells with indices in *DC*.

Let the i^{th} cell be represented by the collection of corresponding indices of segments on each axis, i.e., the i^{th} cell is equivalent to that in $\left(s_{1j_1}, s_{2j_2}, ..., s_{NVj_{NV}}\right)$ where s_{jj_f} is the j_f^{th} segment on the f^{th} variable, $j_f \in [1, NS_f]$, f=1,…,*NV*, and NS_j is the number of segments on j^{th} variable axis.

Then the set of neighbors of the cell $\left(s_{1j_1}, s_{2j_2}, ..., s_{NVj_{NV}}\right)$ can be defined as:

$$\left\{ \left(s_{1h_1}, .., s_{NVh_{NV}}\right) \middle| \begin{matrix} abs(s_{ih_i} - s_{ij_i}) \leq 1, \forall (i, h_i) \\ \text{and } (s_{ih_i} - s_{ij_i}) > 0, \exists (i, h_i) \end{matrix} \right\} \tag{5}$$

The possible neighbors of a cell are described in Fig. 1

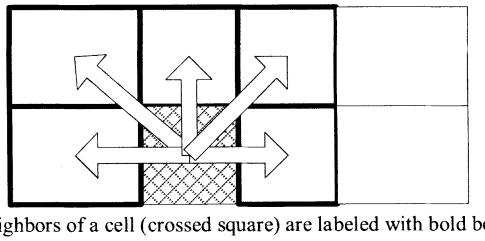

Fig. 1. Neighbors of a cell (crossed square) are labeled with bold borders

Thus, for the i^{th} cell in *SC*, the parameters ε_i and $minPts_i$ are adjusted as follows:

$$\begin{aligned} \varepsilon_i &= \min_{j \in DC \cap NEC_i} \left(\varepsilon_j\right), \forall i \in SC \\ \text{minPts}_i &= \max_{j \in DC \cap NEC_i} \left(\text{minPts}_j\right), \forall i \in SC \end{aligned} \tag{6}$$

where the set of indices of neighboring cells for a given cell with index *I* is defined by

$$NEC_i = \left\{ j \mid j \equiv \left(s_{1h_1}, .., s_{NVh_{NV}}\right), \left(s_{1h_1}, .., s_{NVh_{NV}}\right) \text{ satisfy eqn. 5} \right\} \quad (7)$$

The rationale for selecting the minimum and maximum of ε_i and $minPts_i$ associated with neighboring dense cells will be explained after the last step. If $DC \cap NEC_i = \varnothing$, then ε_i and $minPts_i$ will be adjusted by

$$\varepsilon_i = \left\{ \varepsilon_j \mid ND_j = \max_{i=1,...,|NEC_i|} (ND_i) \right\}, \forall i \in SC$$
$$\text{minPts}_i = \left\{ \text{minPts}_j \mid ND_j = \max_{i=1,...,|NEC_i|} (ND_i) \right\}, \forall i \in SC \quad (8)$$

The next step is to merge the cells with similar epsilon values. Similar to the approach in the CLIQUE method [1], this step is performed by using the same idea of DBSCAN connecting the data vectors in the same cluster. Although CLIQUE connects cells with densities above a threshold, our proposed algorithm connects cells with similar epsilon values to distinguish those regions with different densities.

Definition 1. (directly ε-reachable) A cell i is directly ε-reachable from a cell j with respect to $perc$ if i and j are neighboring cells (eqn (5)) and satisfies the condition

$$1 - \frac{\min(\varepsilon_i, \varepsilon_j)}{\max(\varepsilon_i, \varepsilon_j)} < perc \quad (9)$$

Definition 2. (ε-reachable) A cell i is ε-reachable from a cell j with respect to $perc$ if there is a chain of cells $c_1,\ldots,c_n$, $c_1=i$, $c_n=j$ such that c_{i+1} is directly ε-reachable from c_i.

In order to describe the concept, we provide following example as illustrated in Fig. 2. Let cells 2,3,8 and cells 1,4,5 have epsilon values close to ε_1 and ε_2, respectively, satisfying eqn (9). Cell 6 has an epsilon value close to ε_3 and cell 7 is a neither in SC nor in DC (eqn (2)). Cell (2,3) and (3,8) are directly ε-reachable and hence (2,8) are ε-reachable which implies that cells (2,3,8) can be merged. Cells (1,5) are directly ε-reachable but cell 4 is neither directly ε-reachable or ε-reachable from cell 1 or 5. Thus, cells (1,5) can be merged but cell 4 cannot be merged with any other cell.

After finding the mergable cells with similar ε values, the attributes of the merged cell should be updated. Since the mergable cells have similar epsilon values, the epsilon value does not change significantly. Nevertheless, we take the weighted average of epsilon values of the mergable cells to assign the epsilon value of the merged cell. Then, value of minPts is updated by using eqn (4).

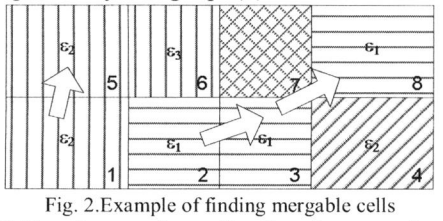

Fig. 2.Example of finding mergable cells

Let NCM be the number of cells after merging the cells with similar epsilon values. Let NC_i be the number of cells that are merged to obtain the i^{th} merged cell. Then the epsilon value of the i^{th} cell, ε_i, and number of data and index of data in the i^{th} cell, ND_i and $indexD_i$, respectively, can be updated as follows:

$$\varepsilon_i = \frac{\sum_{j=1}^{NC_i} ND_j \varepsilon_j}{ND_i}$$
$$ND_i = \sum_{j=1}^{NC_i} ND_j \quad (10)$$
$$indexD_i = \underset{j=1,...,NC_i}{conc} (indexD_j)$$

where $conc$ is the concatenation operator to merge the indices of the mergable cells into one index matrix. CLIQUE [1] and ENCLUS [3] stops at this point and do not differentiate between the regions with different densities.

In the proposed GRIDBSCAN approach, the third stage is to run the DBSCAN algorithm on the data with different density parameters identified by using the procedure above. However, rather than running the main module of DBSCAN for every unprocessed data vector in the data set by using the identified density parameters, we run the main module for every unprocessed data vector in each cell by using the corresponding density parameters. The clusters started by the data vectors in the cell are allowed to be expanded to the data outside the cell. However, the data vectors outside the cell cannot start a new cluster. Thus, if we have NCM cells with different density parameters, we have NCM different results. In order to obtain a single result, the results of the DBSCAN executions are sorted by a confidence parameter value associated to them. The basic idea is to assign a higher confidence to the results of the DBSCAN execution started from a cell with low ε value and high number of data. The confidence value associated to the i^{th} cell with ε_i and ND_i can be calculated as follows:

$$CF_i = \frac{ND_i}{\varepsilon_i^{NV}}, \forall i = 1,..., NCM \quad (11)$$

The third stage of the proposed method will sort the results of multiple DBSCAN execution for each cell from 1 to NCM in descending order with respect to the associated confidence values. The data vectors assigned to a cluster by the DBSCAN execution with the highest confidence value are stored and flagged as assigned. The remaining unassigned data vectors are checked whether they are assigned to a cluster by the DBSCAN execution with the second highest confidence value and so on and so forth. The GRIDBSCAN algorithm is summarized in Algorithm 1. Let λ be the number segments into which each feature space axis is divided. This value is used in identifying the most suitable cells [17] before the second stage, i.e., merging stage, is started.

1. Input $perc$, λ and training data
2. Determine the most suitable cells (or segmentation of each axis by using genetic algorithms with respect to the objective function specified in Reference [17]
3. $\forall i$=1,…, NC DO
3.1 Find cells worthwhile to process further using eqn (2)
3.2 If the cell i is DC or SC then identify ε_i, $minPts_i$, ND_i,

indexD$_i$ by eqns (3) and (4)
4. $\forall i \in SC$ adjust ε_i and *minPts*$_i$ by eqn (6) and (8)
5. Determine the mergable cells from Definitions 1 and 2
6. Update ε_i,*minPts*$_i$, *ND*$_i$, *indexD*$_i$ of the merged cells by eqn (10) and (4)
7. $\forall i$=1,…, *NCM* DO
7.1 Calculate the confidence value for the DBSCAN started from i^{th} merged cell by using eqn (11)
8. Sort the confidence values in descending order
9. $\forall i$=1,…, *NCM* DO
9.1 [clusters, unassigned data vectors] = DBSCAN(ε_i, *minPts*$_i$, *ND*$_i$, *indexD*$_i$, *trainData*)

Algorithm 1. GRIDBSCAN

III. Complexity Analysis

As indicated in Reference [17], the complexity of finding the most suitable segmentation (step 2 in algorithm 1) is O(NG*NP*NC), where NG is the number of generations, NP is the number of candidates in the population, and NC is the number of cells equal to 2^{λ}, where λ is the number segments into which each feature space axis is divided. However, the cells containing no data will be disregarded. Thus, the maximum number of cells is equal to the number of data in the data set, ND. The second step of the algorithm, potentially the most time consuming part, has computational complexity O(NC*ND_i^2) where ND_i is the number of data vectors in the i^{th} cell. The term ND_i^2 occurs because eqn (3) requires measuring the pairwise distances between the data vectors within the cell. NC has an upper limit of ND. Since the algorithm is partially a grid-based algorithm, we expect ND_i to be sufficiently small. Adjusting the cell attributes of sparse cells has a worst case complexity of O(|SC|*(|SC|+|DC|)). Similar to NC, both |SC| and |DC| have an upper limit of ND. However, due to objective function used in genetic algorithm to segment the space, we anticipate a small number of sparse cells. Merging the mergable cells has a complexity of NClogNC as in DBSCAN. While merging the cells, the DBSCAN algorithm is used with respect to ε-reachability rather than density reachability. The second stage of the GRIDBSCAN method has a complexity of O(NCM*NDlogND). This is again due to the fact that the second stage of the proposed method uses the DBSCAN algorithm.

IV. Experimental Results

The proposed algorithm was applied to three artificial benchmark data sets containing arbitrary shaped clusters with different densities. The datasets were created by MATLAB collecting the coordinates of the user mouse clicks. The benchmark datasets are shown in Fig. 3, 4 and 5. The number of data in datasets 1, 2 and 3 were 307, 287 and 397, respectively. In all runs, the input parameters, λ and *perc*, were 10 and 10%, respectively. Since the goal of this example was to detect clusters with different densities, the sensitivity analysis is deferred to a future study.

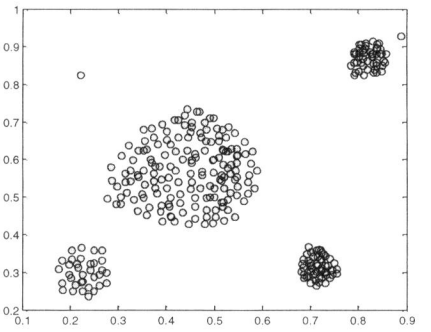

Fig. 3 Dataset 1

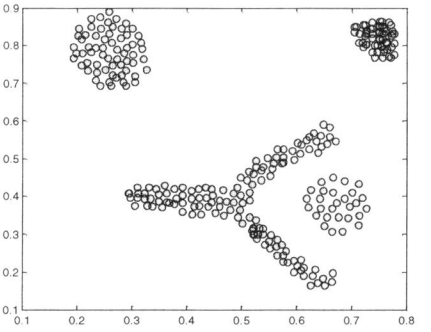

Fig 4. Dataset 2

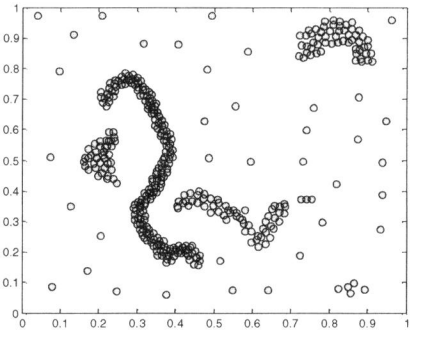

Fig 5. Dataset 3

All steps executed for dataset 1 are shown in Fig.6. Due to space limitations the results for the other two datasets will not be presented.

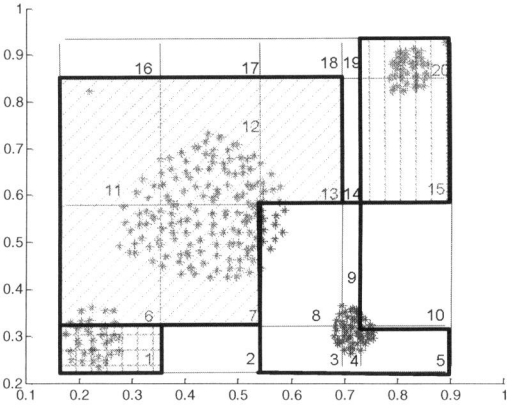

Fig. 6. Steps to cluster the dataset 1

After executing the genetic algorithm, the space was segmented into 20 cells for which the borders are drawn with non-bold lines. Since cells 2, 10, 14, 16, 17, 18 and 19 do not contain enough information, they were not processed further. After adjusting the density parameter values of the remaining sparse cells 3, 5,9, 11, 13 and 15 by using the dense cells 1, 4, 6, 7, 8, 12, and 20, the mergable cells were determined as (3,4,5,8,9), (6,7,11,12,13) and (15,20). After the density parameters were further adjusted in the merged cells, the confidence values of the remaining cells were calculated and sorted in descending order. Starting with density parameter values of the cell with the highest confidence value, DBSCAN was executed the number of times equal to the number of remaining cells after merges. The results for dataset 1, 2 and 3 are shown in Fig. 7, 8 and 9, respectively. The effectiveness of GRIDBSCAN algorithm is measured by its accuracy to assign the data points to correct clusters and time to cluster the data set. The performance indicators are compared with those achieved by DBSCAN. The ε and minPts parameters were the weighted average values of those obtained by GRIDBSCAN for regions with different densities. The weight of each cell was the number of data vectors in the cell. Thus, for datasets 1, 2 and 3, the pairs (ε, minPts) were (0.042,13), (0.03,7), and (0.054,16), respectively. The results are summarized in Table I.

TABLE I
COMPARISON OF GRIDBSCAN AND DBSCAN (DATASET 1, 2, 3)

Dataset	GRIDBSCAN Accuracy (%)	GRIDBSCAN Time (s)	DBSCAN Accuracy (%)	DBSCAN Time(s)
1	97	1.53	76	1.41
2	92	1.43	56	1.27
3	98	2.42	85	2.14

We provided the details of the first step of the proposed method (i.e., identification of most suitable segments using genetic algorithms) in reference [17]. The time reported in Table 1 excludes the computational time required by GA

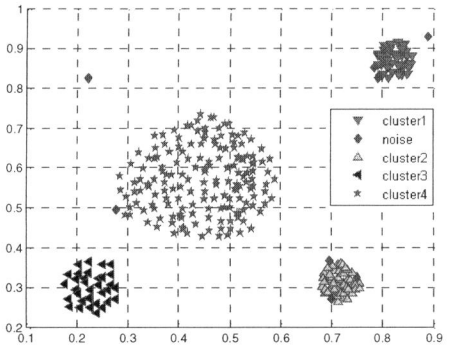

Fig. 7. Results for dataset 1

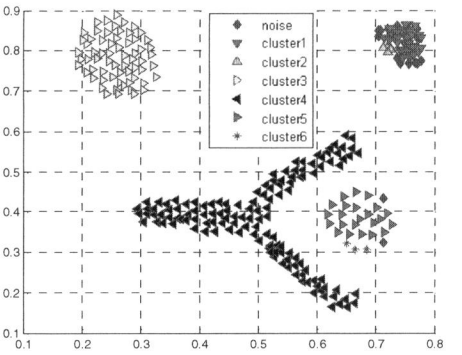

Fig. 8. Results for dataset 2

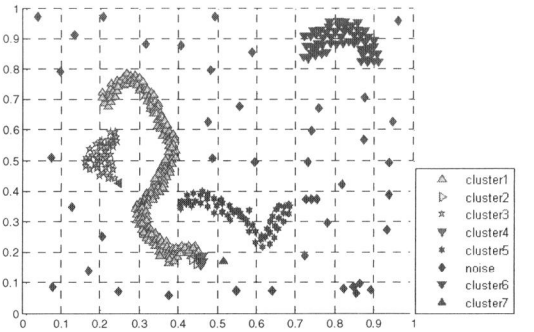

Fig. 9. Results for dataset 3

In order to interpret the results, we shall examine the clustering results for data set 2 in Fig. 8. In this case, the proposed method misassigned some data in cluster 1 as a separate cluster (i.e., cluster 2). This is due to the fact that the density of data in cluster 1 is so high that the data in the outer rim of the cluster is mislabeled as a separate cluster. The same explanation applies to cluster 5 and 6 in Fig. 8. In order to assess the scalability of the GRIDBSCAN method, every data vector in each benchmark dataset has been entered to the datasets 10 times. Hereafter, the new datasets will be called dataset 4, 5 and 6. Hence, the number of data vectors in each data set was increased by a factor of 10. The locations of the data vectors were maintained the same in order to eliminate the effects of the result of the genetic algorithm, i.e., the most suitable segmentation for each dataset was maintained unchanged. However, due to fact that the values *ub* and *lb* changed, it was expected to have changes in the accuracy of the results. A similar procedure was applied to determine the density parameters for DBSCAN execution with datasets 4,5 and 6. Thus, for datasets 1, 2 and 3, the pairs (ε, minPts) were specified as (0.04,119), (0.028,63), and (0.054,161), respectively. The results were summarized in Table II.

TABLE II
COMPARISON OF GRIDBSCAN AND DBSCAN (DATASET 4, 5, 6)

Dataset	GRIDBSCAN Accuracy (%)	GRIDBSCAN Time (s)	DBSCAN Accuracy (%)	DBSCAN Time(s)
4	90	50.66	76	16.5
5	68	64.44	66	12.54
6	99	131.26	89	26.03

It is evident that GRIDBSCAN was superior based on the accuracy of assigning the data vectors to the correct clusters. However, the computational complexity can be an inhibitor of using GRIDBSCAN on very large spatial databases.

V. Conclusion and Future Studies

We have proposed a new clustering algorithm inspired by grid-based and density-based clustering. One of the major difficulties with most density based clustering methods is that they are not effective in the presence of clusters with different densities. The main goal of the method is to address this issue. The proposed algorithm has been applied to three artificial benchmark datasets containing clusters with different densities. The results were compared those obtained by using the DBSCAN algorithm.

In terms of accuracy, GRIDBSCAN was better than DBSCAN. However, in terms of computational complexity, GRIDBSCAN may be expensive because pairwise distances are measured while identifying the density parameters in each cell. Thus, GRIDBSCAN is useful for small and medium size spatial databases but not for large databases.

There are several opportunities for future research. As in DBSCAN, the proposed method has two input parameters that must be provided by the user. First, a sensitivity analysis should be conducted to observe the effects of the change in input parameter values. Second, further research needs to be performed in order to solve the first stage (i.e., the identification of the most suitable segmentation) of the GRIDBSCAN method. Finally, the complexity of the second stage of GRIDBSCAN algorithm, i.e. identification of density parameters in cells with different densities, is due to the fact that pairwise distances are calculated between each data vector pairs within the cell. Sampling methods can be used to decrease the computational complexity of this stage.

VI. Acknowledgment

This research received financial support through an Industrial Partnership Grant from the Natural Sciences and Engineering Research Council of Canada, with contributions from the National Research Council of Canada, and C. P. Loewen Enterprises Ltd., Manitoba, Canada.

References

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications," *Proc. of the ACM SIGMOD'98 International Conference on Management of Data*, Montreal, Canada, 1998, pp. 94-105

[2] M. Ankerst, M.M. Breunig, H.P. Kriegel, and J. Sander, "OPTICS: Ordering Points to Identify the Clustering Structure," *Proc. of the ACM SIGMOD'99 International Conference on Management of Data*, Philadelphia, PA, 1999, pp. 49–60.

[3] C.-H. Cheng, A. W. Fu, and Y. Zhang, " Entropy-Based Subspace Clustering for Mining Numerical Data," *Proc. of the 5th International Conference on Knowledge Discovery and Data Mining*, San Diego, CA, 1999, pp. 84-93.

[4] J. L. Devore, *Probability and Statistics for Engineering and Sciences*, Duxbury Press, New York, NY, 1995.

[5] M. Ester, H..P. Krigel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining*, Portland, WA, 1996, pp. 226-231.

[6] D.H. Fisher, "Knowledge Acquisition Via Incremental Conceptual Clustering," *Machine Learning*, vol. 2, pp. 139-172, 1987

[7] J. Han and M. Kamber, *Data Mining: Concepts and Techniques.* San Diego, CA: Morgan Kaufmann, 2001, pp. 347-348.

[8] A. Hinneburg and D.A. Keim, "An Efficient Approach to Clustering in Multimedia Databases with Noise," *Proc. of the 4th International Conference on Knowledge Discovery and Data Mining*, New York, 1998, pp. 58-65.

[9] G. Karypis, E.-H. Han and V. Kumar, "Chameleon: Hierarchical Clustering using Dynamic Modeling," *Computer*, vol. 32, pp. 68-75, Aug 1999.

[10] L. Kaufman and P.J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley, 1990.

[11] T. Kohonen, *Self-Organization and Associative Memory*. New York, NY, Springer-Verlag, 1988.

[12] S. Milgram, "The Small World Problem," *Psychology Today,* pp 60-67, May 1967.

[13] R. Ng and J. Han, "Efficient and Effective Clustering Method for Spatial Data Mining," *Proc. of the International Conference on Very Large Data Bases*, Santiago, Chile, 1994, pp. 144-155.

[14] L. Parsons, E. Haque, and H. Liu, "Subspace Clustering for High Dimensional Data: A Review," *ACM SIGKDD Explorations Newsletter*, vol. 6, pp. 90-105, June 2004

[15] G. Sheikholeslami, S. Chatterjee, and A. Zhang, "WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases," *Proc. of the 24th International Conference on Very Large Databases,* San Francisco, CA, 1998, pp. 428-439.

[16] G. Sudipto, R. Rastogi and K. Shim, "CURE: An Efficient Clustering Algorithm for Large Databases," *Proc. of the 1998 ACM SIGMOD International Conference on Management of Data*, Seattle, WA, 1998, pp. 73-84.

[17] O. Uncu, W. A. Gruver, D. B. Kotak, and D. Sabaz, "Application of GA to the GRIDBSCAN Clustering Algorithm," *Technical Report TR219,* Intelligent/Distributed Enterprise Automation Laboratory, School of Engineering Science, Simon Fraser University, Burnaby, BC Canada, January 2006.

[18] X. Xu, M. Ester, H.P. Kriegel, and J. Sander, "A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases," *Proc. of the 14th International Conference on Data Engineering*, Orlando, FL, 1998, pp. 324-331.