A novel genetic algorithm for automatic clustering

Gautam Garai [a], B.B. Chaudhuri [b,*]

[a] *Computer Division, Saha Institute of Nuclear Physics, 1/AF Bidhannagar, Kolkata 700 064 India*
[b] *Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, 203 B.T. Road, Kolkata 700 035 India*

Received 10 December 2002; received in revised form 8 September 2003

Abstract

In this paper we have presented a new genetically guided algorithm for solving the clustering problem. The proposed Genetic Clustering Algorithm is basically a two-phase process. At the first phase the original data set is decomposed into a number of fragmented clusters in order to spread the GA search process at the latter phase over the entire space. At the second phase Hierarchical Cluster Merging Algorithm (HCMA) is used. The HCMA is an iterative genetic algorithm based approach that combines some of the fragmented clusters into complete *k*-cluster. The algorithm contains another component called Adjacent Cluster Checking Algorithm (ACCA). This technique is used for testing adjacency of two segmented clusters so that they can be merged into one cluster. The performance of the algorithm has been demonstrated on several data sets consisting of multiple clusters and it is compared with some well-known clustering methods.

Keywords: Clustering; Genetic algorithm; Splitting; Merging; Optimization

1. Introduction

Cluster analysis is an effective tool in scientific inquiry. Clustering technique groups a set of data in *d*-dimensional feature space to maximize the similarity within the clusters and to minimize the similarity between two different clusters. Various clustering methods have been developed in exploratory data analysis, image segmentation, pattern recognition and with some added knowledge, may be used for classification as well (Anderberg, 1973; Bhuyan et al., 1991; Devijver and Kittler, 1982; Dubes and Jain, 1980; Fu, 1980; Hartigan, 1975; Tou and Gonzalez, 1974). These methods can broadly be classified into two categories, namely, *hierarchical* and *non-hierarchical* clustering. The former approach can further be divided into *agglomerative* and *divisive* methods. In the agglomerative approach, two most similar clusters at each level are merged together and the merged clusters remain intact at all higher levels. In the divisive method, the complete dataset is initially considered as a single cluster which is then divided into smaller clusters at each level of hierarchy depending on the properties of the data.

* Corresponding author. Tel.: +91-33-577-2088; fax: +91-33-577-6680/2577-6680.

E-mail addresses: bbc@isical.ac.in, bbc@www.isical.ac.in (B.B. Chaudhuri).

Several non-hierarchical methods have been proposed for clustering data. Among them, the k-means algorithm is perhaps the most popular one. It is a simple iterative hill-climbing algorithm where the solution obtained depends on the initial clustering. This algorithm and its variant have been successfully employed in many practical clustering problems (Selim and Ismail, 1984). The algorithm, however, fails to converge to the true minimum under certain conditions. A branch-and-bound algorithm has been proposed by Koontz et al. (1975) to find clustering globally although it requires much computation time. In the single-linkage algorithm based on a probability model, Ling (1973) tried to define clusters using two indices measuring compactness and relative isolation.

Clustering techniques have been used in a wide range of disciplines. For example, in psychiatry, Levine et al. (1969) used clustering to develop a classification of mental depression. In market research, clustering algorithm is used to identify homogeneous sets of test markets (Green et al., 1967). Levrat et al. (1992) developed fuzzy clustering algorithm to segment an image in pattern recognition problems. Funk et al. (1987) used the clustering algorithm as a method to knowledge acquisition for expert system assisted diagnosis. Moreover, as an example of clustering for time incremental data, new data are acquired at each exploration step for clustering by discovery (Chaudhuri, 1994). Yin and Chen (1994) has presented the nearest-neighbor algorithm based on conventional neighborhood clustering method. In this technique, the threshold of distance for grouping objects is a required predefined parameter which the users must choose judicially.

Apart from the classical algorithms, several soft computing approaches are also successfully used in solving the clustering problem. A simulated annealing algorithm is presented by Klein and Dubes (1989) for finding a globally optimum solution under some conditions. The evolutionary strategies have also been explored for solving clustering problems (Babu and Murty, 1994) with the a priori knowledge of the number of clusters by the user. Similarly, Alippi and Cucchiara (1992) and Bezdek et al. (1994) have applied GAs for their clustering methods where the required number of clusters is known a priori. Various adaptation techniques have been used to enable the GAs to cluster and to enhance their performance (Krovi, 1991). Hall et al. (1999) has used his genetically guided approach in optimizing the hard and fuzzy c-means functionals for cluster analysis.

The clustering problem is defined as the problem of classifying a collection of homogeneous data points into a set of natural clusters without any a priori knowledge. When clustering data points, it is necessary to amalgamate all the variables into a single index of similarity. Due to good performance as stochastic search procedure Genetic Algorithm is used for cluster representation (Murthy and Chowdhury, 1996). Several *split-and-merge* techniques have also been employed in clustering objects with maximum similarity. Some researchers have used GA based on split-and-merge method in defining clusters. Tseng and Yang (2001) have proposed a scheme where the dataset is first split and then the smaller clusters are merged using GA. However, the algorithm fails if one cluster is confined partially or fully within another cluster. DBScan (Ester, 1996) is another algorithm based on split-and-merge procedure where the fragmented clusters of arbitrary shapes are merged if the cluster density measured beforehand is uniform. Guha et al. (1998) have proposed their method called CURE for finding clusters of arbitrary shape and various sizes in the absence of noise. The Chameleon (Karypis et al., 1999) clustering method uses the graph partitioning scheme while splitting and min-cut bisection method for determining the most similar clusters for merging process. Another interesting and useful clustering technique is the k-windows algorithm which exploits a well-known spatial data structure, namely *the range tree*. It achieves high quality clustering results with low time complexity compared to other well-known clustering algorithms. However, it is not easily applicable on high dimensional data due to super-linear space requirements for the range tree. An improved algorithm has been proposed by Alevizos et al. (2002) where a modified orthogonal range search technique has been employed.

In this paper we present a Genetically based Clustering Algorithm (GCA) which is basically a two-stage *split-and-merge* algorithm for finding the clusters. Initially, the entire data set is decomposed into a reasonably large number of fragmented clusters. These clusters are then automatically combined using a Hierarchical Cluster Merging Algorithm (HCMA) in several cycles until the given k clusters are obtained. The cluster merging process is actually based on genetic algorithm. In the first cycle the algorithm merges some clusters. The process is reinvoked and continued for defining k clusters. The decomposition process is used to distribute the search of the HCMA over the entire search space. In the experiment the users can also easily specify the input parameters for finding k clusters. The prespecified parameters given by the user are r, k and T_{d}. The input k is the required number of clusters while r is the radius of the imaginary circular region which encloses the data points of the segmented clusters. The entire data is decomposed into a number of smaller clusters depending on r. T_{d} denotes the threshold of density difference between pair of clusters to be merged.

We have tested our algorithm on several data sets in d-dimensional feature space, R^d. The data sets in our experiment consist of a wide variety of clusters. They are widely separated or closely spaced or a combination of both. We have also tested data sets where one cluster is confined within another and also clusters are present with noisy data. In such a situation Adjacent Cluster Checking Algorithm (ACCA) associated with HCMA is employed to separate the desired number of clusters accurately. The method GCA has also been tested on popular and well-known *Iris data* (Duda and Hart, 1973; Fisher, 1936).

The remaining sections are arranged as follows. The basic concept of the classical genetic algorithm along with its algorithmic implementation is presented in Section 2. The proposed method GCA is described in Section 3. It provides the description of the algorithms CDA, HCMA and ACCA. Section 4 presents the experimental results on various data sets in two and multi-feature space. A comparison of the proposed method with the other well-known clustering algorithms is also discussed in this section. Some discussions and the conclusion of the paper are included in Section 5.

2. Basic concept of Classical Genetic Algorithm

Genetic Algorithm (GA) is a class of stochastic search procedure capable of adaptive and robust search over a wide range of search space. The process is inspired by the Darwinian's principle of survival of the fittest individuals and of natural selection. The technique was first introduced by Holland (1975) for use in adaptive systems. It was then employed by several researchers in solving various optimization problems effectively and efficiently.

The search procedure starts with the initialization of a few parameters which may/may not be modified in course of the search. The algorithm passes through three basic phases iteratively, namely, the reproduction phase, the crossover phase and the mutation phase. The detailed operation at each phase is lucidly described in (Goldberg, 1989). The Classical Genetic Algorithm (CGA) can be described as follows.

1. Generate randomly the initial population of μ individuals and let $g = 1$. Initialize δ and η, where δ is the crossover probability and η is the mutation probability.
2. Evaluate the fitness score for each individual $\boldsymbol{p}_i$, $\forall i \in \{1, \ldots, \mu\}$ of the population based on the objective function, $f(\boldsymbol{p}_i)$ where $\boldsymbol{p}_i$'s are objective variables.
3. Select a pair of individuals $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ at random depending on their fitness values (using roulette wheel method) from the population of μ individuals.
4. Conduct crossover between the chosen individuals, $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ with δ and mutate each bit of each parents with mutation probability, η.
5. Each pair of parents, $(\boldsymbol{p}_a, \boldsymbol{p}_b)$ creates a pair of new individuals called offsprings $(\boldsymbol{p}'_a, \boldsymbol{p}'_b)$. The offsprings thus generate a pool of individuals, $\boldsymbol{p}'_j$, $\forall j \in \{1, \ldots, \mu\}$ as a population of next generation.

6. Terminate the process if the stopping criterion, $g > G_{\max}$ is satisfied where $G_{\max}$ is the maximum number of generations. Otherwise, $g = g + 1$ and go to step 2.

3. Clustering with Genetic Algorithm

Consider a set of n vectors $\mathscr{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ to be clustered into k groups of homogeneous data. Each $\boldsymbol{x}_i \in R^d$ is a feature vector consisting of d real valued measurements describing the features of the object represented by $\boldsymbol{x}_i$. The features can be length, breath, color, etc.

The clustering approach proposed here mainly composes of two steps. The first one is called Cluster Decomposition Algorithm (CDA) and the second one is Hierarchical Cluster Merging Algorithm (HCMA).

3.1. Splitting of clusters with CDA

The process of CDA first decomposes the entire data set, $\mathscr{X}$ into m (the value of m varies for different data sets) groups of clusters. Each cluster B_i, $\forall i \in \{1, 2, \dots, m\}$ is the collection of a few similar objects among $\boldsymbol{x}_j$'s, $\forall j \in \{1, 2, \dots, n\}$ where $k < m \leqslant n$. Thus, m number of clusters are initially generated by CDA from the single cluster, B (the entire data set, $\mathscr{X}$) such that

$$B = B_1 \cup B_2 \cup \cdots \cup B_m$$

and

$$B_i \cap B_j = \phi$$

where $i, j \in \{1, 2, \dots, m\}$, $k < m \leqslant n$ and $i \neq j$ and

$$n = \sum_{i=1}^{m} |B_i|$$

where $|B_i|$ stands for the size of the cluster, B_i i.e., the number of data points in it.

The progress of the process is shown in Fig. 1. Now the CDA is implemented in the following way.

Step 1. For each object $\boldsymbol{x}_i$, $\forall i \in \{1, 2, \dots, n\}$ find the distance, $d_{\min}$ between $\boldsymbol{x}_i$ and its nearest neighbor in the data set $\mathscr{X}$.

$$d_{\min}(\boldsymbol{x}_i) = \min_{i \neq j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \tag{1}$$

where $j \in \{1, 2, \dots, n\}$ and $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| = \sum_{l=1}^{d} \sqrt{(\boldsymbol{x}_{il} - \boldsymbol{x}_{jl})^2}$.

Step 2. Compute d_{av}, the average of the minimum distances, $d_{\min}(\boldsymbol{x}_i)$, $\forall i \in \{1, 2, \dots, n\}$ using Eq. (1) as follows:

$$d_{\mathrm{av}} = \frac{1}{n} \sum_{i=1}^{n} d_{\min}(\boldsymbol{x}_i) \tag{2}$$

Step 3. Consider x_i, $\forall i \in \{1, 2, \dots, n\}$ as the center of a circular region with radius r which encloses a group of objects from $\mathscr{X}$. Evaluate the radius r to create fragmented clusters B_p's, $\forall p \in \{1, 2, \dots, m\}$ containing the objects of the circular region of radius r in the following way:

$$r = u * d_{\mathrm{av}} \tag{3}$$

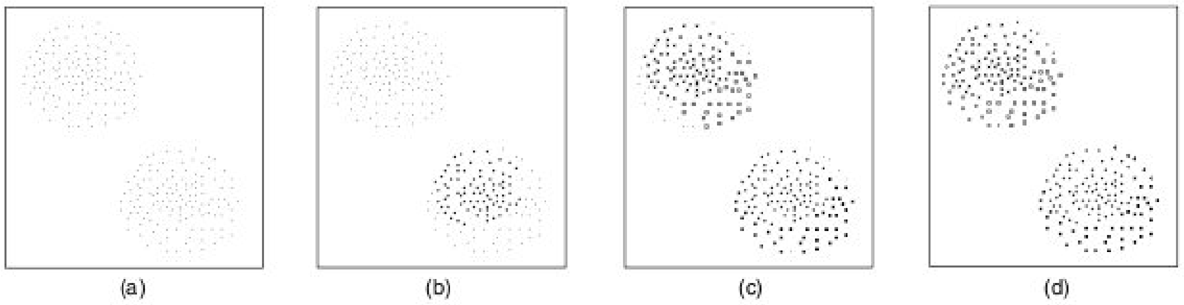

Fig. 1. The progress of CDA: (a) the original dataset with two clusters; (b) creation of one fragmented cluster after the use of CDA; (c) five fragmented clusters at the intermediate state of CDA and (d) creation of 10 fragmented clusters at the end of CDA.

where the real value of u lies between 2 and 4.

Step 4. Set $p = 1$.

Step 5. Extract B_p and modify the data set $\mathscr{X}$ such that $\mathscr{X} = |\mathscr{X} - B_p|$.

Step 6. Terminate the algorithm if $|\mathscr{X}| = \phi$. Otherwise, $p = p + 1$ and go to step 5.

The time complexity of CDA is analyzed as follows. Since the size of dataset is n, step 1 takes $O(n^2)$ time to calculate minimum distance between pair of points. Step 2 takes $O(n)$ time to evaluate the average of the min-distances. Step 5 and step 6 are executed for $O(m)$ time to decompose the dataset and $m \ll n$. Therefore, CDA ultimately takes $O(n^2)$ time.

3.2. Cluster merging with HCMA

After the completion of the CDA, the HCMA starts processing at the second stage to merge the fragmented clusters, B_i's, $\forall i \in \{1,\ldots,m\}$ which are homogeneous in nature. At this step the genetic algorithm is invoked over B_i's considering each B_i is a single object. The GA starts in the conventional way with a population of μ individuals where each individual, $\boldsymbol{p}_j$, $\forall j \in \{1,\ldots,\mu\}$ is a string of m bits created randomly and uniformly from $\{0,1\}$. In each $\boldsymbol{p}_j$ if the ith bit is 1, it indicates the presence of cluster, B_i. If the corresponding bit is 0, it denotes the absence of B_i.

The GA is now invoked on a population (a set of individuals) and terminated after $G_{\max}$ iterations. This is the completion of one GA cycle and it will be termed as a *GA cycle* or only *cycle* in the rest of this paper. In our proposed approach several such cycles are completed for obtaining desired k clusters. In each cycle some of m B_i's are merged to create m' clusters. Thus the process gradually merges a few clusters in each cycle and the merged clusters remain in the same cluster in all latter cycles. The progress of the merging process is illustrated in Fig. 2.

The algorithm, HCMA consists of all three phases of CGA. At the first phase of GA two individuals, $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$, are chosen randomly from the pool of μ individuals. They are then mutually crossed over with the crossover probability, δ using single point crossover operation to generate two offsprings, $\boldsymbol{p}'_a$ and $\boldsymbol{p}'_b$, respectively. The third operation mutation is performed bitwise over $\boldsymbol{p}'_a$ and $\boldsymbol{p}'_b$ with the adaptive mutation probability, η_{adap} to produce $\boldsymbol{p}''_a$ and $\boldsymbol{p}''_b$, respectively. The value of η_{adap} is evaluated as follows:

$$\eta_{\text{adap}} = \eta_0 * t \tag{4}$$

where t stands for the tth cycle and η_0 is the initial mutation probability.

The fitness value of each individual, $\boldsymbol{p}''_i$, $\forall i \in \{1,\ldots,\mu\}$ are now calculated and the above three phases are continued for $G_{\max}$ generations. At the termination of each GA cycle, some among m clusters are merged to have m' ($m' < m$) number of clusters. The merging process of the clusters proceeds as follows.

In the second stage the HCMA starts with a set of strings $\{B_1, B_2, \ldots, B_m\}$ where each B_i is considered to be an object. Some (say, m_0) of these B_i's

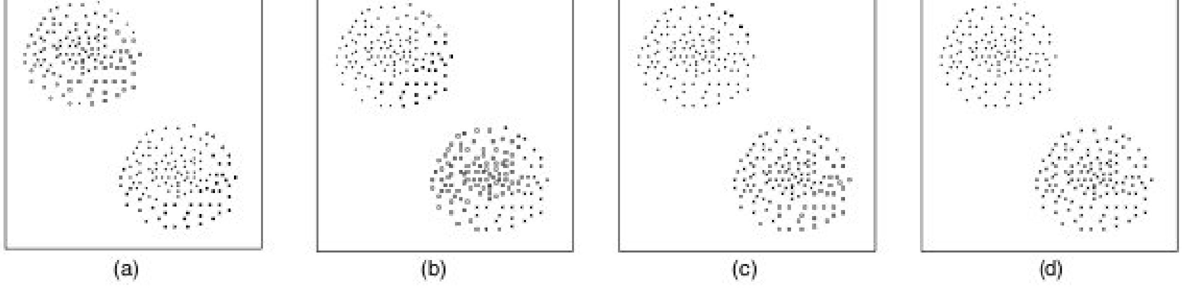

Fig. 2. The progress of HCMA without ACCA: (a) fragmented clusters before the start of HCMA (output of CDA); (b) merging of two-fragmented clusters after the use of HCMA; (c) merging of six-fragmented clusters at the intermediate state of HCMA and (d) isolated two clusters at the end of HCMA.

are 0 and some of them (say, m') are 1. Let the B_i's which are 0 be represented as $B^0 = \{B^0_1, B^0_2, \ldots, B^0_{m_0}\}$ and B_i's which are 1 be defined as $B^1 = \{B^1_1, B^1_2, \ldots, B^1_{m'}\}$. Now, initially each B^1_i, $\forall i \in \{1, \ldots, m'\}$ is assigned as the only candidate of each cluster C_i, $\forall i \in \{1, \ldots, m'\}$ where $k \leqslant m' < m$ in the merging process. It then starts searching an object B_e from B^0_j's, $\forall j \in \{1, \ldots, m_0\}$ so that it satisfies the relation

$$B_e = \min \|\text{seed}(B^1_i) - \text{seed}(B^0_j)\| \tag{5}$$

where $i \in \{1, 2, \ldots, m'\}$.

Once an object, B_e in B^0_j's is selected as a candidate of cluster C_i, $\forall i \in \{1, \ldots, m'\}$ for merging, it will be excluded from the list of B^0. Thus, the fragmented cluster merging process is continued for a string, B until all B^0_j's, $\forall j \in \{1, \ldots, m_0\}$ are exhausted to generate clusters C_i's, $\forall i \in \{1, \ldots, m'\}$ where $k \leqslant m' < m$. Now each cluster C_i, $\forall i \in \{1, \ldots, m'\}$ is represented as

$$C_i = \bigcup_{l=1}^{q} B_l \tag{6}$$

where $1 \leqslant q < m$ and $m = m_0 + m'$.

Let the seed of the fragmented cluster B_i be β_i, $\forall i \in \{1, \ldots, m\}$ and that of the newly generated cluster C_j be S_j, $\forall j \in \{1, \ldots, m'\}$. Now the center S_j of each C_j is computed with the following equation:

$$S_j = \frac{1}{q} \sum_{l=1}^{q} \beta_l \tag{7}$$

where $B_l \subset C_j$ and $1 \leqslant q < m$.

In HCMA each individual $\boldsymbol{p}_i$, $\forall i \in \{1, \ldots, \mu\}$ is a string of $\{B_1, B_2, \ldots, B_m\}$. Now the fittest individual is extracted from the pool of μ individuals to create a new pool of equal size for the next generation. It is, therefore, required to evaluate the fitness function, $\mathscr{F}(\boldsymbol{p}_i)$ of each individual $\boldsymbol{p}_i$. The function is, however, dependent on two important functions of cluster C_α, $\forall \alpha \in \{1, \ldots, m'\}$, namely, $D_{\text{intra}}(C_\alpha)$ and $D_{\text{inter}}(C_\alpha)$. The fitness function $D_{\text{intra}}(C_\alpha)$ represents the intra-distance in the cluster C_α. On the other hand, $D_{\text{inter}}(C_\alpha)$ stands for the inter-distance in C_α's. Now, the above two functions are defined by the following two equations:

$$D_{\text{inter}}(C_\alpha) = \max_{\alpha \neq \gamma} \|S_\alpha - S_\gamma\| * |C_\alpha| \tag{8}$$

where $\alpha, \gamma \in \{1, 2, \ldots, m'\}$ and,

$$D_{\text{intra}}(C_\alpha) = \sum_{\gamma=1}^{q} \left(\|S_\alpha - \beta_\gamma\| * \frac{|B_\gamma|}{|\mathscr{X}|} \right) \tag{9}$$

where $\alpha \in \{1, 2, \ldots, m'\}$ and $B_\gamma \subset C_\alpha$.

We can define the fitness function, $\mathscr{F}(\boldsymbol{p}_i)$ of a string or an individual as follows:

$$\mathscr{F}(\boldsymbol{p}_i) = \sum_{\alpha=1}^{m'} D_{\text{inter}}(C_\alpha) - \sum_{\alpha=1}^{m'} D_{\text{intra}}(C_\alpha) * |B^1| \tag{10}$$

where $i \in \{1, 2, \ldots, \mu\}$.

The entire process with three phases of CGA is repeated for $G_{\max}$ times. After that the clusters C_α's, $\forall \alpha \in \{1, \ldots, m'\}$ are considered as B_i's and $m = m'$ for the next cycle of GA. The process is continued until the predefined number of clusters k ($k > 1$) is found.

The time complexity of the merging GA process is as follows. μ denotes the population size and m is the length of the chromosome in the GA. Maximum $O(m^2)$ time is required to find the nearest cluster for each chromosome. The calculation of fitness function needs $O(\mu m^2)$ time for entire population. $G_{\max}$ is the maximum number of generations to run each GA cycle and $T_{\max}$ is the maximum number of GA cycles to find k clusters. Therefore, the time complexity of the merging algorithm is $O(T_{\max} G_{\max} \mu m^2)$.

3.3. Adjacency checking between two fragmented clusters

The clusters are generally identified distinctly using the above two algorithms, CDA and HCMA. However, sometimes it is absolutely necessary to exercise ACC algorithm. The procedure is eventually used along with HCMA if one cluster is confined fully or partly within another cluster and if clusters are present in noisy data (see Figs. 7–11). In such data sets it is required to test the adjacency properties of the candidate cluster B^0_j, $\forall j \in \{1, \ldots, m\}$ if it fulfills the merging conditions with the member cluster B^1_i of C_i, $\forall i \in \{1, \ldots, m'\}$. At the first phase the splitting process is restricted for the application of ACCA so that each smaller clusters must contain at least four data points. The ACCA uses two thresholds T_b and T_d for deciding merging of pair of clusters. The threshold of boundary points T_b is always 4 in our experiment and the threshold of data density difference for a pair of fragmented clusters to be merged is T_d which depends on the test data set. The value of T_d is normally selected measuring the density of a few (≈5% of total subclusters) randomly chosen sub-clusters after the completion of CDA. The algorithm is implemented in the following way for verifying adjacency of any two sub-clusters which are primarily selected for merging by HCMA.

Step 1. Define suitably the value of the radius r'. [r' is equal or closed to r computed in Eq. (3)].
Step 2. Select two fragmented clusters, B_f^0 and B_g^1 from B^0 and B^1, respectively which satisfy the merging condition (see Eq. (5)).
Step 3. Count the number of boundary points of B_f^0 and B_g^1 which resides within radius r'. Let it be N_b and the object density of B_f^0 and B_g^1 be $\mathscr{D}_f^0$ and $\mathscr{D}_g^1$, respectively.
Step 4. If $N_b \geqslant T_b$ and $\mathrm{abs}(\mathscr{D}_f^0 - \mathscr{D}_g^1) \leqslant T_d$ then B_f^0 and B_g^1 are adjacent to each other. Include B_f^0 as the member of C_i, $\forall i \in \{1, \ldots, m'\}$.
Step 5. Terminate the algorithm.

4. Experimental results

In our experiment we have considered various types of data sets. The description of data sets is provided in Sections 4.2 and 4.3. The objects of data sets in Figs. 3–11 are in R^2 feature space. We have also considered the popular Iris data in R^4 feature space. Of Figs. 3–8 three data sets comprise of two clusters, two data sets contain three clusters and a single data set is made of five clusters. Figs. 7–9 illustrate the situation where one cluster is fully or partly enclosed by the other cluster. Figs. 10 and 11 are more difficult data sets. Fig. 10 consists of six clusters, of which two elliptical shaped clusters are connected to each other by a thin dense line. Fig. 11 is a collection of two clusters in the presence of noise.

4.1. Parameter setting

In all of our experiments the same self-adaptive method (see Eq. (4)) and the same population size, $\mu = 50$ are used for HCMA. The initial population is generated uniformly at random depending on the number of clusters, B_i, $\forall i \in \{1, \ldots, m\}$. The number m is inversely proportional to the value of r. The parameter u is a real number between 2 and 4. The number of ultimate clusters, k is pre-specified by the user. We have tested each data set for 30 runs and in each run the crossover probability δ and the initial mutation probability η_0 lie in the range [0.5–0.9] and [0.002–0.005], respectively. The GA is employed in cluster merging process and iterated for $G_{\max} = 100$ times in each cycle. The user specified threshold of boundary points T_b is

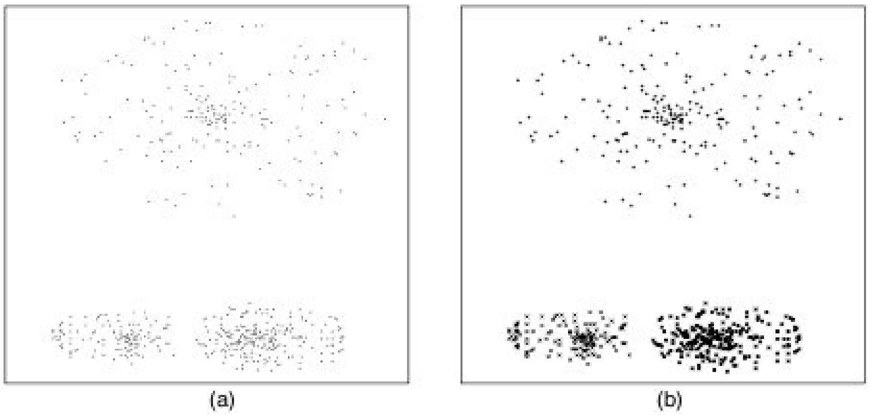

Fig. 3. (a) The original dataset with three clusters before the application of GCA and (b) isolated three clusters after the completion of GCA.

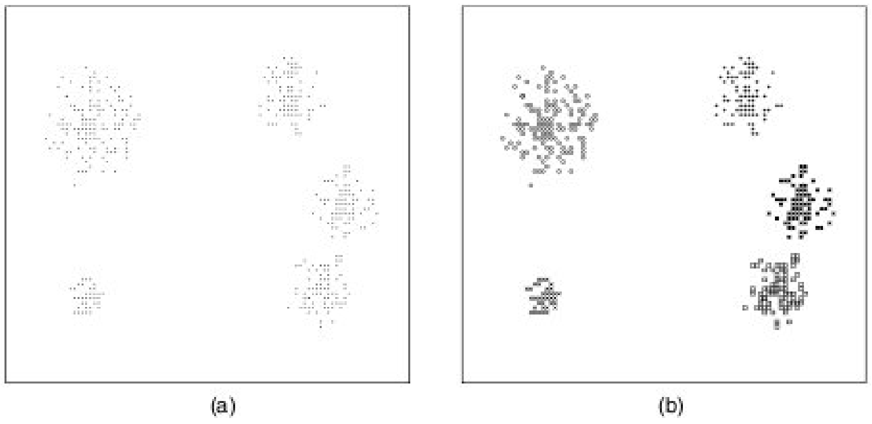

Fig. 4. (a) The original dataset with five clusters before the application of GCA and (b) isolated five clusters after the completion of GCA.

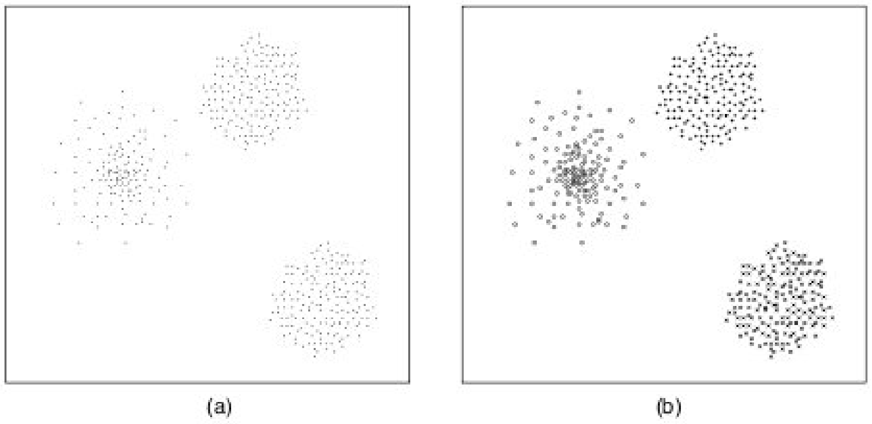

Fig. 5. (a) The original dataset with three clusters before the application of GCA and (b) isolated three clusters after the completion of GCA.

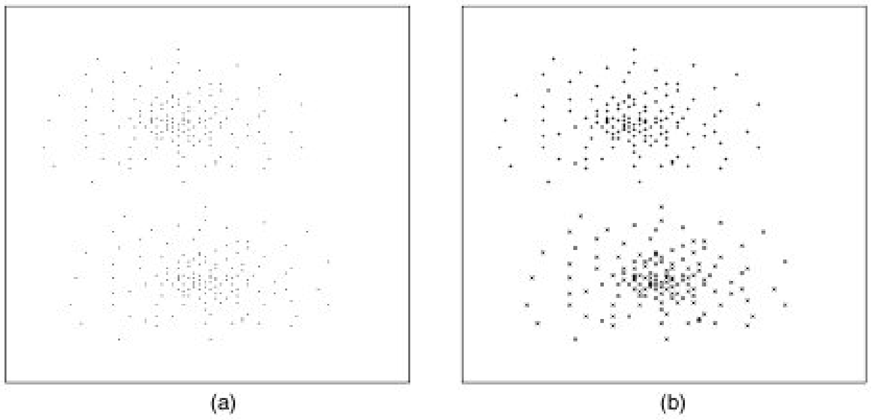

Fig. 6. (a) The original dataset with two clusters before the application of GCA and (b) isolated two clusters after the completion of GCA.

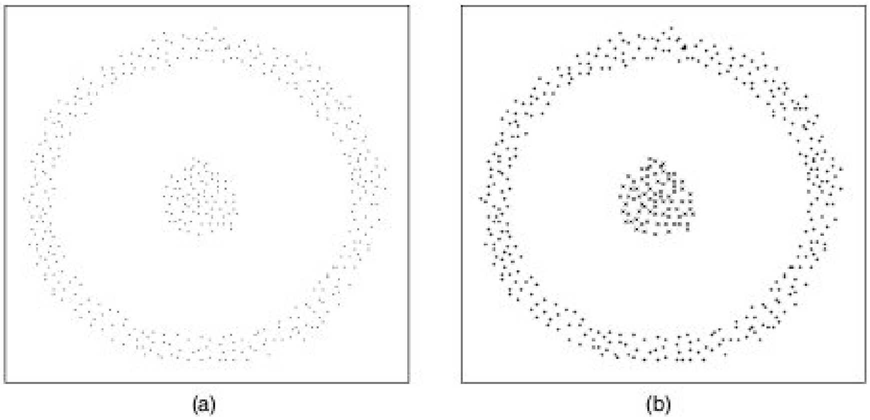

Fig. 7. (a) The original dataset with two clusters before the application of GCA and (b) isolated two clusters after the completion of GCA.

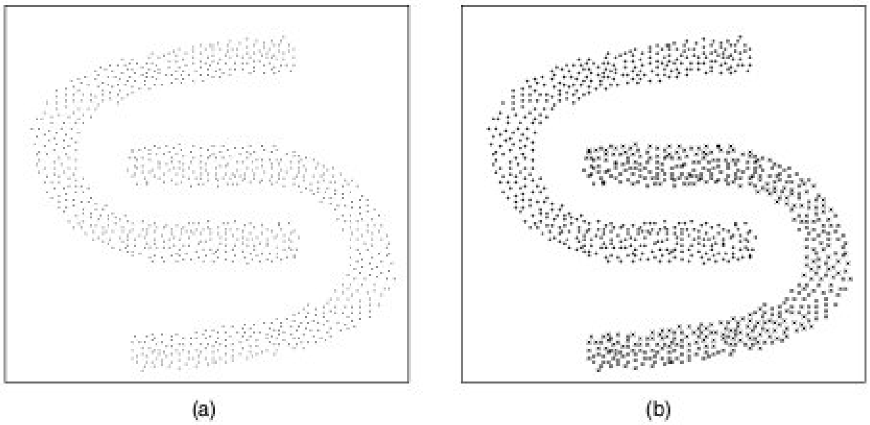

Fig. 8. (a) The original dataset with two clusters before the application of GCA and (b) isolated two clusters after the completion of GCA.

always 4 and that of data density difference T_d needed for cluster merging is chosen as 0.4. The computation time of the algorithm is increased with the increase of the value of user defined parameters, either μ or G_{max} or both. On the other, the increase of the value of r decreases the computation time as the string length or chromosome size, m is reduced.

4.2. Cluster partitioning in R^2 feature space

In our test we have studied nine data sets as samples for checking GCA in R^2 feature space. Among them, both Figs. 3 and 5 consist of three clusters, although the nature of clusters are different. In both figures two clusters are located closely whereas the third one is placed wide apart compared to the position of the former two clusters. However, in Fig. 3 the density of closely placed clusters are non-uniform. On the other hand, the third one is three times the size of either of the two clusters and the data points are denser at the center of the cluster. The density gradually reduces towards the cluster boundary. Fig. 3(a) shows the condition of clusters before using GCA and Fig. 3(b) illustrates how the clusters are isolated

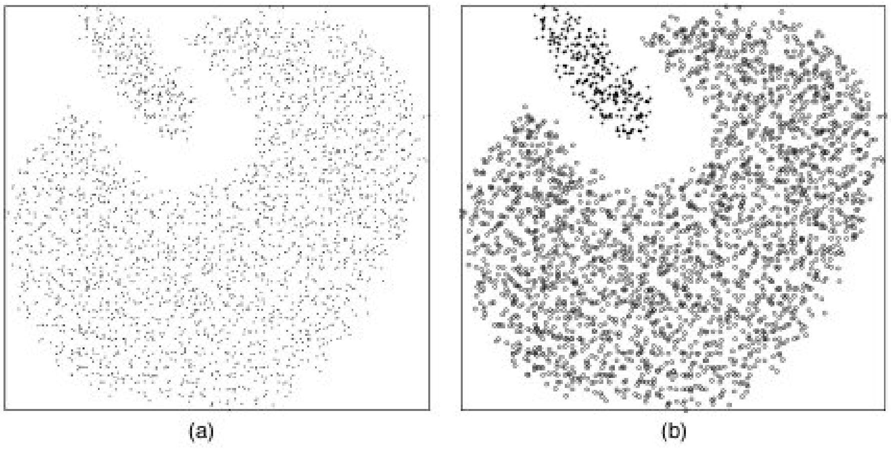
Fig. 9. (a) The original dataset with two clusters before the application of GCA and (b) isolated two clusters after the completion of GCA.

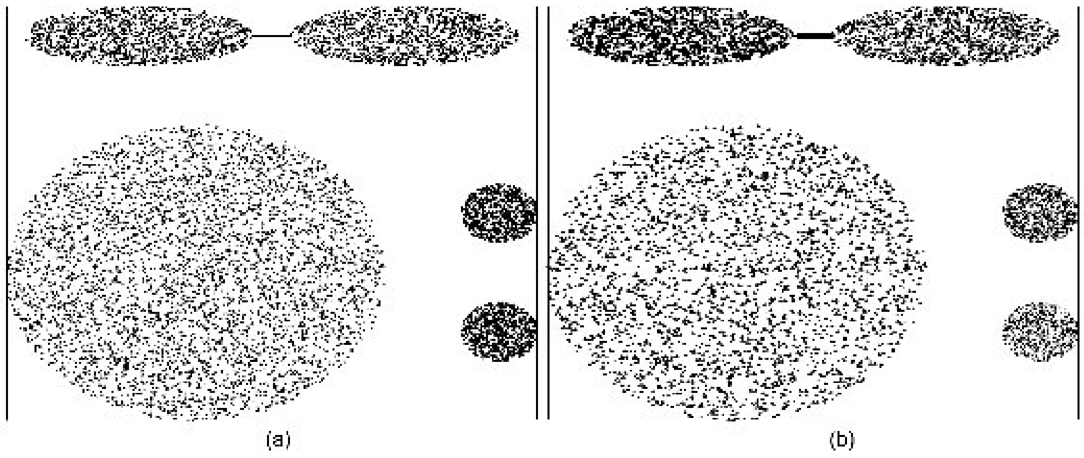
Fig. 10. (a) The original dataset with six clusters before the application of GCA and (b) isolated six clusters after the completion of GCA.

after using the proposed method. In this case ACCA is not employed since any of the clusters is not enclosed by another cluster. From our test runs it is noted that for this dataset the algorithm fails approximately 15% of total runs to isolate closely located pair of clusters.

On the other hand, in Fig. 5 the density of two closely located clusters are different. One of them is uniformly dense and the other has density tapering away from the center. The third cluster is of uniform density. However, all three clusters are of equal size. Fig. 5(a) and (b) depict the situations, respectively, before and after the application of GCA. All three clusters are separated properly in all test runs.

The data set in Fig. 4(a) comprises of five number of clusters. All of them are denser near the center and lighter towards the boundary. Three clusters are equal in size and two of them are close to each other. In all 30 test runs, the five clusters have been correctly isolated after using GCA as shown in Fig. 4(b).

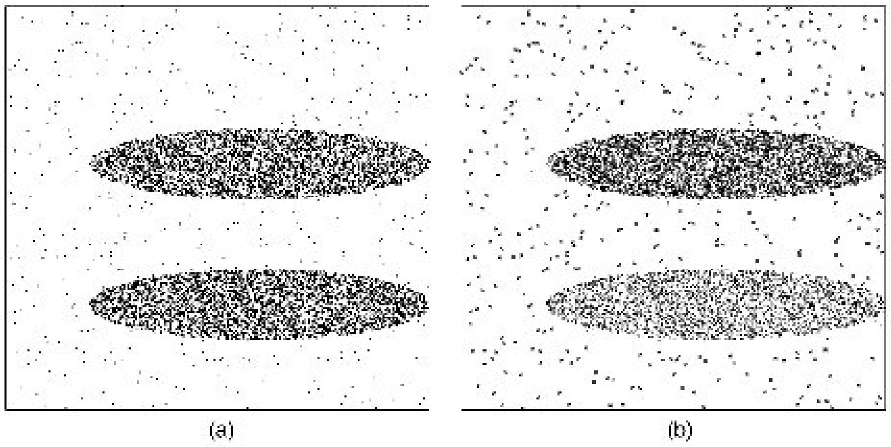

Fig. 11. (a) The original dataset with two clusters and noise before the application of GCA and (b) isolated two genuine clusters and separated noise after the completion of GCA.

Fig. 6 shows an important and interesting data in our experiment where both clusters almost overlap each other. Moreover, they are equal in size and density. The data points in both the cluster follow Gaussian distribution in R^2 space. Two clusters are so closely located that they appear like a single cluster. The presented algorithm can distinctly identify the clusters, as shown in Fig. 6(b) in all test runs.

Another two data sets (Figs. 7 and 8) are different from the four data sets explained earlier. Here, each figure comprises of two clusters where one cluster is either fully or partly enclosed by the other. The density of data points of both clusters in both figures is almost uniform. If CDA followed by HCMA is employed here, part of one cluster is merged with another cluster. In Fig. 7 approximately 60% of data points of the big circle are merged with the small circular cluster. Similar situation happens for the data in Fig. 8. However, the application of ACCA associated with HCMA on the data sets can perfectly isolate the two clusters as shown in Figs. 7(b) and 8(b). Identical results have been obtained in all test runs.

Figs. 9–11 consist of special type of clusters. Among them, Fig. 9 contains two clusters of arbitrary shapes and different sizes. The data density is also random in nature. One of the clusters is too small compared to the other and the smaller one is partially within the larger cluster. Finally, the clusters are represented properly as shown in Fig. 9(b) using HCMA along with ACCA after decomposing the data set into several relatively smaller sub-clusters with CDA.

Fig. 10(a) is a collection of six clusters which are of different size and density. Two of the six clusters are of identical elliptical shape and are connected to each other with a string of dense data points in 2-D space. The set of these two clusters can be considered as a single cluster as they are not disjoint. They are, however, shape-wise three clusters and have been defined accurately in our experiment. Here, the third cluster is the string of points connecting those two elliptical shaped clusters. Two of the remaining three clusters in Fig. 10(a) are almost equal in shape and density while the third one is the largest cluster among all with lesser density compared to the other. Fig. 10(b) illustrates all six correctly separated clusters (the string of points as one cluster) after the application of HCMA in combination with the ACCA.

The data set in Fig. 11(a) is absolutely different from all other data sets used in our experiment. The data set is also very special in nature as it comprises of two clusters of identical shape and different densities in the presence of random noise scattered all over the feature space. Fig. 11(b) shows two genuine clusters identified perfectly and the noise is represented as the third cluster. At the first stage the data set is segmented into smaller sub-clusters and then HCMA with ACCA is used for merging the sub-clusters to represent the genuine clusters in presence of noise. After the proper identification of the genuine two clusters it is noticed that the noise in the space is clustered locally in smaller arbitrary shape due to the inherent property of random noise. All such smaller clusters of noise are merged together to combine them into one cluster of noise as shown in Fig. 11(b).

All the data sets in our experiment have been tested for at least 30 times. Although all the figures depict that the clusters are identified clearly, it is observed during test runs that sometimes the algorithm fails for some data sets. One such data set is shown in Fig. 3(a). Here, the algorithm wrongly identified clusters in approximately 15% of the total test runs. The data sets in Figs. 10(a) and 11(a) are quite different from the remaining data sets. During testing it is also viewed that the algorithm wrongly represented the pair of elliptical shaped clusters along with the dense thin line (Fig. 10(a)) for nearly 20% of test runs. Further, a portion of neighboring noise data of the genuine clusters in Fig. 11(a) has been improperly merged with either of two genuine clusters for about 15% of cases. For the remaining data sets the proposed method has always identified clusters accurately in all test runs.

The experimental results of various well-known spatial clustering algorithms in (Karypis et al., 1999) on some standard data sets illustrate that each method has some restrictions for which each one may sometimes fail to cluster data accurately. Density-Based Spatial Clustering of Applications with noise (DBScan) is a popular clustering algorithm for identifying clusters of arbitrary shapes. It represents a cluster which has to be a maximum set of density-connected points. As a result every centroid in a cluster must have at least a minimum number of points within a given radius. The smaller adjacent clusters are then merged by traversing a path of density connected points. Thus, the split-and-merge process can cluster data well. The algorithm is usually applicable if the arbitrary shaped clusters are of uniform density and the cluster density is determined beforehand.

Researchers have proposed another split-and-merge based clustering algorithm, Clustering Using Representative (CURE) to remedy the drawbacks of DBScan method. In CURE, a cluster is initially defined by selecting a constant number of well-scattered points and compressing them towards the clusters centroid according to a compression factor. During merging process it measures the similarity of the closest pair of boundary points belonging to two different clusters to be merged. Thus the method can find clusters of arbitrary shapes and different sizes as it represents each cluster with multiple representative points. The technique, however, fails in merging clusters when the underlying data do not follow the assumed model or when the genuine clusters are present in noisy data.

The algorithm named Chameleon represents clusters even in the presence of noise. The process has alleviated the major limitations of both earlier mentioned methods as it concentrates on the closeness as well as the interconnectivity of the clusters selected for merging. Here, the data set is first segmented into a number of relatively smaller sub-clusters using a graph-partitioning scheme based on *k*-nearest-neighbor graph approach. The merging of clusters is dependent on the similarity between pairs of clusters. The similarity is measured considering the relative interconnectivity (RI) and relative closeness (RC) of pair of clusters to be merged. These measurements of cluster-pair are based on the min-cut bisection of a connected graph. Finally, the clusters are merged which satisfy the user-specified thresholds of RI and RC. The technique becomes costly since the selection of min-cut bisection of graph is a time-consuming process. The computation time of the process might also increase further with the increase of feature space dimension.

The limitations of the well-known methods and a comparison among them have been discussed by experimenting all processes on some synthetic sample databases (Karypis et al., 1999). For our experiment we have prepared similar kind of data sets. It helps to show that on identical data the GCA can represent the clusters accurately. The technique is also not restricted to the shape as well as the density of the clusters. The experimental results on various data sets justify that the proposed method can be compared with some of the best clustering algorithms.

4.3. Cluster separation in Iris data

Iris data with four features is one of the most popular databases in the pattern recognition literature. The data set contains three classes named as Iris Setosa (Class A), Iris Versicolor (Class B) and Iris Virginica (Class C) of 50 instances each where each class refers to a type of Iris plant. The feature attributes of the data are *Sepal length*, *Sepal width*, *Petal length* and *Petal width*. Among the three classes, one is linearly separable from the other two classes. However, other two are not linearly separable from each other.

Confusion Matrix for Iris Classification

	Classified as Class A	Classified as Class B	Classified as Class C
Actual Class A	50	0	0
Actual Class B	0	40	10
Actual Class C	0	0	50

Initially we have used CDA on Iris data with $r = 0.3$, 0.4 and 0.5 in different runs (among total 30 runs) to create a number of fragmented clusters B_i's, $\forall i \in \{1,\ldots,m\}$. The HCMA is now employed on B_i's until the number of final clusters are 3. One cluster has always been separated clearly from other two clusters in all 30 runs of our experiment. However, the other two clusters are not perfectly separable. The best result in our experiment is described above by a *Confusion Matrix*.

5. Discussion and conclusion

A genetically guided clustering approach has been described to represent clusters. The proposed GCA is composed of two algorithms. One of them is called CDA and the other is named HCMA. The algorithm is simple to implement and it terminates after several GA cycles when k clusters are found. It can also identify clusters accurately when one cluster is either partly or fully enclosed by another cluster as in Figs. 7–9. Moreover, the process does not require too complex calculation for setting the parameter values. The GCA can identify the required number of k clusters without trial and error process by multiple manual adjustment of the inputs to the algorithm. The parameters to be provided by the user are the radius r of imaginary circle for segmenting the original data set, $\mathcal{X}$ into m smaller (fragmented) clusters, the final number of clusters k and the threshold of data density difference T_d for merging pair of clusters.

In the first stage CDA is applied on the original data set, $\mathcal{X}$ and simply splits it into a group of relatively smaller sub-clusters B_i's, $\forall i \in \{1,\ldots,m\}$ depending on the value of r. This process promotes the GA search over the entire feature space. The second approach of the GCA at the next step is HCMA which is ideally the application of GA in several cycles. The GA is applied on fragmented clusters B_i's, $\forall i \in \{1,\ldots,m\}$ and after the completion of $G_{\max}$ iterations some clusters are merged at each cycle. The string or chromosome size of the GA is m which depends on the value of r. The GA is again invoked over the existing fragmented clusters and the process is continued until k clusters are represented.

In some cases HCMA is exercised in association with ACCA if a test data set contains a cluster entirely or partly confined by another cluster. The purpose of the algorithm is to restrict the merging of two clusters iff they are not adjacent to each other. The examples of such data sets are illustrated in Figs. 7–9. The proposed method is equally applicable to find clusters of arbitrary shapes and different sizes as in Figs. 7 and 9. The data set in presence of noise as depicted in Fig. 11 can also cluster data by HCMA with ACCA.

We have tested our approach on several data sets in R^2 and R^4 feature space. The data sets contain the clusters of various sizes and with varying positions. Some of them are closely positioned and some are widely separated from one another. The popular four dimensional Iris data was considered as a test data set. The performance of the process on the Iris data is quite encouraging. Also, the proposed split-and-merge based method has been compared with some well-known split-and-merge based clustering algorithms CURE, DBScan as well as Chameleon.

Acknowledgements

The authors wish to thank anonymous referees for their precious comments and constructive criticism on the previous version of this article. We would also like to thank U. Bhattacharya and U. Garain of CVPR unit, ISI and Rahul Banerjee of C&MB division, SINP for their help.

References

Alevizos, P., Boutsinas, B., Tasoulis, D., Vrahatis, M.N., 2002. Improving the orthogonal range search K-windows algorithm. In: Proc. 14th IEEE Int. Conf. on Tools with Artificial Intell., pp. 239–245.

Alippi, C., Cucchiara, R., 1992. Cluster partitioning in image analysis and classification: A genetic algorithm approach. In: Proc. CompEuro 1992 on Comput. Systems Software Eng.. IEEE Computer Society Press, Silver Spring, MD, pp. 139–144.

Anderberg, M.R., 1973. Cluster Analysis for Application. Academic Press, New York.

Babu, G.P., Murty, M.N., 1994. Clustering with evolution strategies. Pattern Recognition 27 (2), 321–329.

Bezdek, J.C., Boggavarapn, S., Hall, L.O., Bensaid, A., 1994. Genetic algorithm guided clustering. In: Proc. First IEEE Conf. on Evolution. Comput. IEEE World Congress on Computat. Intell.. IEEE, New York, pp. 34–39.

Bhuyan, J.N., Raghavan, V.V., Elayavalli, V.K., 1991. Genetic algorithm for clustering with an ordered representation. In: Belew, R.K., Booker, L.B. (Eds.), Proc. Int. Conf. on Genetic Algorithm '91. Morgan Kaufman, San Mateo, CA, pp. 408–415.

Chaudhuri, B.B., 1994. Dynamic clustering for time incremental data. Pattern Recognition Lett. 15, 27–34.

Devijver, P.A., Kittler, J., 1982. Pattern Recognition—A Statistical Approach. Prentice-Hall, London.

Dubes, R., Jain, A.K., 1980. Clustering Methodology in Exploratory Data Analysis. Academic Press, New York.

Duda, R.O., Hart, P.E., 1973. Pattern Classification and Scene Analysis, 218.

Ester, M., 1996. A Density-Based Algorithm for discovering clustering in large spatial databases with noise. In: Proc. 2nd Int. Conf. on Knowledge Discovery Data Mining. AAAI Press, Menlo Park, CA, pp. 226–231.

Fisher, R.A., 1936. The use of multiple measurements in taxonomic problems. Annu. Eugen. 7 (Part II), 179–188.

Fu, K.S., 1980. Communication and Cybernetics: Digital Pattern Recognition. Springer, Berlin.

Funk, M., Appel, R.D., Roch, Ch., Hochstrasser, D., Pellegrini, Ch., Muller, A.F., 1987. Knowledge acquisition in expert system assisted diagnosis: A machine learning approach. In: Proc. AIME-87. Springer Verlag, Berlin, pp. 99–103.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, New York.

Green, P.E., Frank, R.E., Robinson, P.J., 1967. Cluster analysis in test market selection. Manage. Sci. 13 (8), 387–400 (Series B).

Guha, S., Rastogi, R., Shim, K., 1998. CURE: An efficient clustering algorithm for large databases. In: Proc. ACM SIGMOD Int. Conf. Management Data. ACM Press, New York, pp. 73–84.

Hall, L.O., Ozyurt, I.B., Bezdek, J.C., 1999. Clustering with a genetically optimized approach. IEEE Trans. Evolut. Comput. 3 (2), 103–112.

Hartigan, J.A., 1975. Clustering Algorithms. Wiley, New York.

Holland, J., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI.

Karypis, G., Han, E.-H., Kumar, V., 1999. Chameleon: Hierarchical clustering using dynamic modeling. IEEE Comput. (Aug), 68–75.

Klein, R.W., Dubes, R.C., 1989. Experiments in projection and clustering by simulated annealing. Pattern Recognition 22, 213–220.

Koontz, W.L., Narendra, P.M., Fukunaga, K., 1975. A branch and bound clustering algorithm. IEEE Trans. Comput. c-24, 908–915.

Krovi, R., 1991. Genetic algorithms for clustering: A preliminary investigation. In: Milutinovic, V., Shriver, B.D., Numaker Jr., J.F., Sprague Jr., R.H. (Eds.), Proc. Twenty-eighth Hawaii Int. Conf. on System Sci.. IEEE Computer Society Press, Silver Spring, MD, pp. 540–544.

Levine, S., Pilowski, I., Boulton, D.M., 1969. The classification of depression by numerical taxonomy. Brit. J. Psychiat. 115, 937–945.

Levrat, E., Bombardier, V., Lamotte, M., Bremont, J., 1992. Multi-level image segmentation using fuzzy clustering and local membership variations detection. In: Proc. IEEE Int. Conf. on Fuzzy Systems. IEEE, New York, pp. 221–228.

Ling, R.F., 1973. A probability theory of cluster analysis. J. Amer. Statist. Assoc. 68, 159–164.

Murthy, C.A., Chowdhury, N., 1996. In search of optimal clusters using genetic algorithm. Pattern Recognition Lett. 17, 825–832.

Selim, S.Z., Ismail, M.A., 1984. K-means-type algorithm: Generalized convergence theorem and characterization of local optimality. IEEE Trans. Pattern Anal. Mach. Intell. 6, 81–87.

Tou, J.T., Gonzalez, R.C., 1974. Pattern Recognition Principles. Addition-Wesley, Reading, MA.

Tseng, L.Y., Yang, S.B., 2001. A genetic approach to the automatic clustering problem. Pattern Recognition 34, 415–424.

Yin, P.Y., Chen, L.H., 1994. Anew non-iterative approach for clustering. Pattern Recognition Lett. 15, 125–133.