A Point Symmetry Based Clustering Technique for Automatic Evolution of Clusters

Sanghamitra Bandyopadhyay, *Senior Member, IEEE,* Sriparna Saha, *Student Member, IEEE*

***Abstract*— In this article, a new symmetry based genetic clustering algorithm is proposed which automatically evolves the number of clusters as well as the proper partitioning from a data set. Strings comprise both real numbers and the don't care symbol in order to encode a variable number of clusters. Here, assignment of points to different clusters are done based on a point symmetry based distance rather than the Euclidean distance. A newly proposed point symmetry based cluster validity index, *Sym*-index, is used as a measure of the validity of the corresponding partitioning. The algorithm is therefore able to detect both convex and non-convex clusters irrespective of their sizes and shapes as long as they possess the symmetry property. Kd-tree based nearest neighbor search is used to reduce the complexity of computing point symmetry based distance. A proof on the convergence property of variable string length genetic algorithm with point symmetry based distance clustering (VGAPS-clustering) technique is also provided. The effectiveness of VGAPS-clustering compared to variable string length Genetic K-means algorithm (GCUK-clustering) and one recently developed weighted sum validity function based hybrid niching genetic algorithm (HNGA-clustering) is demonstrated for nine artificial and five real-life data sets.**

***Index Terms*— Unsupervised classification, cluster validity index, symmetry, point symmetry based distance, Kd tree, Genetic Algorithms, Real encoding**

I. Introduction

Clustering [1] is a core problem in data mining with innumerable applications spanning many fields. In order to mathematically identify clusters in a data set, it is usually necessary to first define a measure of similarity or proximity which will establish a rule for assigning patterns to the domain of a particular cluster centroid. The measure of similarity is usually data dependent. It may be noted that one of the basic features of shapes and objects is symmetry. Symmetry is considered as a pre-attentive feature which enhances recognition and reconstruction of shapes and objects [2]. As symmetry is so common in the natural world, it can be assumed that some kind of symmetry exists in the clusters also. Based on this, a new point symmetry (PS) based distance d_{ps} (PS-distance) is developed in [3]. This overcomes the limitations of an earlier point symmetry distance proposed in [4]. For reducing the complexity of computing the PS-distance, use of Kd-tree [5] is incorporated in [3]. This proposed distance is then utilized to develop a genetic algorithm based clustering technique, GAPS, where the number of clusters is assumed to be known *apriori* [3].

Determining the appropriate number of clusters from a given data set is an important consideration in clustering. For this purpose, and also to validate the obtained partitioning, several cluster validity indices have been proposed in the literature. The measure of validity of the clusters should be such that it will be able to impose an ordering of the clusters in terms of its goodness. The classical approach of determining the number of clusters is to apply a given clustering algorithm for a range of *K* values and to evaluate a certain validity function of the resulting partitioning in each case [6], [7], [8], [9], [10], [11], [12], [13], [14]. The partitioning exhibiting the optimal validity is chosen as the true partitioning. This method for searching an optimal number of cluster number depends on the selected clustering algorithm, whose performance may depend on the initial cluster centers. Furthermore, most of the validity measures usually assume a certain geometrical structure in the cluster shapes. But if several different structures exist in the same data set, these have often been found to fail.

Since the global optimum of the validity functions would correspond to the most "valid" solutions with respect to the functions, stochastic clustering algorithms based on Genetic Algorithms (GAs) have been reported to optimize the validity functions to determine the cluster number and partitioning of a data set simultaneously [9], [15], [16]. Other than evaluating the static clusters generated by a specific clustering algorithm, the validity functions in these approaches are used as clustering objective functions for computing the fitness, which guides the evolution to search for the "valid" solution. However, Simple GA (SGA) [17] or its variants are used as the genetic clustering techniques in [9], [15], [16]. In [18], a function called Weighted Sum Validity Function (WSVF), which is a weighted sum of the several normalized validity functions, is used for optimization along with a Hybrid Niching Genetic Algorithm (HNGA) to automatically evolve the proper number of clusters from a given data set. Within this HNGA, a niching method is developed to prevent premature convergence by preserving both the diversity of the population with respect to the number of clusters encoded in the individuals and the diversity of the subpopulation with the same number of clusters during the search. In [19], a multi-objective genetic approach is used for clustering where several validity functions are simultaneously optimized.

In the above mentioned genetic clustering techniques for automatic evolution of clusters, assignment of points to different clusters are done in the lines of *K*-means clustering algorithm. Consequently, all these approaches are only able to find compact hyperspherical, equisized and convex clusters like those detected by the *K*-means algorithm [20]. If clusters of different geometric shapes are present in the same data set, the above methods will not be able to find all of them perfectly. This article presents an attempt in this direction. Here we define a cluster validity index named *Sym*-index (symmetry based cluster validity index) that uses a new definition of the point symmetry (PS) distance (d_{ps}) that is able to detect clusters of any shape and size as long as they possess the symmetry property.

In this paper, a variable string length GA (VGA) based clus-

Sriparna Saha is the corresponding author. Authors are with the Machine Intelligence Unit, Indian Statistical Institute, 203, B.T.Road, Kolkata-700108, India, Email: {sanghami,sriparna_r}@isical.ac.in

tering method is used as the underlying segmentation technique. Here assignment of points to different clusters is done based on the PS distance. The *Sym*-index is used as the optimizing criterion. The characteristic features of the proposed clustering technique, referred to as VGAPS-clustering, are as follows. Use of variable string length GA allows the encoding of a variable number of clusters. The *Sym*-index, used as the fitness function, provides the most approximate partitioning even when the number of clusters, K, is varied. Again use of GA enables the algorithm to come out of local optima, a typical problem associated with local search methods like the *K*-means (note that optimizing *Sym*-index is not inherent to a GA framework. Any other optimization technique, such as Simulated Annealing [21] may be used rather). Finally use of the PS-distance enables the evolution of clusters of any shape and size as long as they possess the symmetry property. Using finite Markov chain theory, a convergence proof of VGAPS-clustering to a globally optimal partition is also established. In [22], a very preliminary version of fuzzy variable string length GA clustering technique with point symmetry distance is proposed. But no convergence proof nor any detailed comparative results were presented in [22]. In this paper the effectiveness of the proposed VGAPS-clustering for evolving the appropriate partitioning of a dataset is demonstrated on nine artificial and five real-life data sets having different characteristics. The performance of VGAPS-clustering is compared with those of GCUK-clustering [16] and the recently proposed HNGA-clustering [18].

II. A New Definition of the Point Symmetry Distance

In this section, a new PS distance [3], $d_{ps}(\overline{x},\overline{c})$, associated with point $\overline{x}$ with respect to a center $\overline{c}$ is described. As shown in [3], $d_{ps}(\overline{x},\overline{c})$ is able to overcome some serious limitations of an earlier PS distance [4]. Let a point be $\overline{x}$. The symmetrical (reflected) point of $\overline{x}$ with respect to a particular centre $\overline{c}$ is $2\times\overline{c}-\overline{x}$. Let us denote this by $\overline{x}^*$. Let $knear$ unique nearest neighbors of $\overline{x}^*$ be at Euclidean distances of d_is, $i=1,2,\ldots knear$. Then

$$\begin{aligned} d_{ps}(\overline{x},\overline{c}) &= d_{sym}(\overline{x},\overline{c})\times d_e(\overline{x},\overline{c}), &(1)\\ &= \frac{\sum_{i=1}^{knear} d_i}{knear}\times d_e(\overline{x},\overline{c}), &(2)\end{aligned}$$

where $d_e(\overline{x},\overline{c})$ is the Euclidean distance between the point $\overline{x}$ and $\overline{c}$ and $d_{sym}(\overline{x},\overline{c})$ is a symmetry measure of $\overline{x}$ with respect to $\overline{c}$. It can be seen from Equation 2 that $knear$ cannot be chosen equal to 1, since if $\overline{x}^*$ exists in the data set then $d_{ps}(\overline{x},\overline{c})=0$ and hence there will be no impact of the Euclidean distance. On the contrary, large values of $knear$ may not be suitable because it may underestimate the amount of symmetry of a point with respect to a particular cluster center. Here $knear$ is chosen equal to 2. It may be noted that the proper value of $knear$ largely depends on the distribution of the data set. A fixed value of $knear$ may have many drawbacks. For instance, for very large clusters (with too many points), 2 neighbors may not be enough as it is very likely that a few neighbors would have a distance close to zero. On the other hand, clusters with too few points are more likely to be scattered, and the distance of the two neighbors may be too large. Thus a proper choice of *knear* is an important issue that needs to be addressed in the future.

Note that $d_{ps}(\overline{x},\overline{c})$, which is a non-metric, is a way of measuring the amount of point symmetry between a point and a cluster center, rather than the distance like any Minkowski distance.

The benefits of using several neighbors instead of just one in Equation 2 are as follows.

1) Here since the average distance between $\overline{x}^*$ and its $knear$ unique nearest neighbors have been taken, this term will never be equal to 0, and the effect of $d_e(\overline{x},\overline{c})$, the Euclidean distance, will always be considered. Note that if only the nearest neighbor of $\overline{x}^*$ is considered and this happens to coincide with $\overline{x}^*$, then this term will be 0, making the distance insensitive to $d_e(\overline{x},\overline{c})$. This in turn would indicate that if a point is marginally more symmetrical to a far off cluster than to a very close one, it would be assigned to the farthest cluster. This often leads to undesirable results as demonstrated in [3].
2) Considering the $knear$ nearest neighbors in the computation of d_{ps} makes the PS-distance more robust and noise resistant. From an intuitive point of view, if this term is less, then the likelihood that $\overline{x}$ is symmetrical with respect to $\overline{c}$ increases. This is not the case when only the first nearest neighbor is considered which could mislead the method in noisy situations.

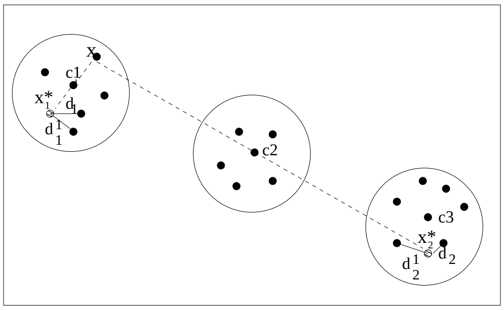


Fig. 1. Example of a data set having some symmetrical interclusters

Definition 1: The Euclidean distance difference (EDD) property is defined as follows:
Let $\overline{x}$ be a data point, $\overline{c}_1$ and $\overline{c}_2$ be two cluster centers, and θ be a distance measure. Let $\theta_1=\theta(\overline{x},\overline{c}_1)$, $\theta_2=\theta(\overline{x},\overline{c}_2)$, $d_{e_1}=d_e(\overline{x},\overline{c}_1)$ and $d_{e_2}=d_e(\overline{x},\overline{c}_2)$. Then θ is said to satisfy EDD property if for $\frac{\theta_1}{\theta_2}<\frac{d_{e2}}{d_{e1}}$, point $\overline{x}$ is assigned to $\overline{c}_1$, otherwise it is assigned to $\overline{c}_2$.

Observation 1: *The proposed symmetry measure satisfies the Euclidean distance difference property.*

Proof : Let us assume that there are two clusters, having cluster centers $\overline{c}_1$ and $\overline{c}_2$. Let $\overline{x}$ be a particular data point. Let the *knear* nearest neighbors of the reflected point of $\overline{x}$ with respect to center $\overline{c}_1$ be at distances of d_i^1, $i=1,\ldots,knear$. Then $d_{ps}(\overline{x},\overline{c}_1)=d_{sym}(\overline{x},\overline{c}_1)\times d_{e1}=\frac{\sum_{i=1}^{knear} d_i^1}{knear}\times d_{e1}$, where d_{e1} is the Euclidean distance between $\overline{x}$ and $\overline{c}_1$. Let the *knear* nearest neighbors of the reflected point of $\overline{x}$ with respect to center $\overline{c}_2$ be at distances of d_i^2, $i=1,\ldots,knear$. Hence, $d_{ps}(\overline{x},\overline{c}_2)=d_{sym}(\overline{x},\overline{c}_2)\times d_{e2}=\frac{\sum_{i=1}^{knear} d_i^2}{knear}\times d_{e2}$, where d_{e2} is the Euclidean distance between $\overline{x}$ and $\overline{c}_2$. Now in order to preserve the EDD property given that $\frac{d_{sym}(\overline{x},\overline{c}_1)}{d_{sym}(\overline{x},\overline{c}_2)}<\frac{d_{e2}}{d_{e1}}$, the point $\overline{x}$ is assigned to center $\overline{c}_1$. Point $\overline{x}$ is assigned to cluster of $\overline{c}_1$ if $d_{ps}(\overline{x},\overline{c}_1)<d_{ps}(\overline{x},\overline{c}_2)$. This indicates

that

$$\begin{aligned} \frac{\sum_{i=1}^{knear} d_i^1}{knear} \times d_{e1} &< \frac{\sum_{i=1}^{knear} d_i^2}{knear} \times d_{e2} \quad (3) \\ \Longrightarrow \frac{\frac{\sum_{i=1}^{knear} d_i^1}{knear}}{\frac{\sum_{i=1}^{knear} d_i^2}{knear}} &< \frac{d_{e2}}{d_{e1}} \quad (4) \\ \Longrightarrow \frac{d_{sym}(\overline{x},\overline{c}_1)}{d_{sym}(\overline{x},\overline{c}_2)} &< \frac{d_{e2}}{d_{e1}}. \quad (5) \end{aligned}$$

It therefore becomes evident that the d_{sym} satisfies the EDD property defined in Definition 1.

Definition 2: If two clusters are symmetrical to each other with respect to a third cluster center, then these clusters are called “symmetrical interclusters”.

Observation 2: *The proposed d_{ps} measure is able to detect the symmetrical interclusters properly*.

Proof: In Figure 1, the first and the third clusters are “symmetrical interclusters” with respect to the middle one. As explained in the above example, though there exists a symmetrical point of $\overline{x}$ with respect to cluster center $\overline{c}_2$, but $\overline{x}$ is assigned to the first cluster as the newly developed d_{ps} distance satisfies the EDD property. As a result, the three clusters present in Figure 1 are identified properly. Thus it is proved that the proposed point symmetry based distance is able to detect symmetrical interclusters properly.

It is evident that the symmetrical distance computation is very time consuming because it involves the computation of the nearest neighbors. Computation of $d_{ps}(\overline{x_i},\overline{c})$ is of complexity $O(nD)$, where D is the dimension of the data set and n is the total number of points present in the data set. Hence for K clusters, the time complexity of computing point symmetry distance between all points to different clusters is $O(n^2KD)$. In order to reduce the computational complexity, an approximate nearest neighbor search using the Kd-tree approach is adopted in this article.

A. Kd-tree Based Nearest Neighbor Computation

A K-dimensional tree, or Kd-tree is a space-partitioning data structure for organizing points in a K-dimensional space. ANN (Approximate Nearest Neighbor) is a library written in C++ [23], which supports data structures and algorithms for both exact and approximate nearest neighbor searching in arbitrarily high dimensions. In this article ANN is used to find exact d_is, where $i = 1, \ldots, knear$, in Equation 2 efficiently. The ANN library implements Kd-tree data structure.

The function performing the *k*-nearest neighbor search in ANN is given a query point *q*, a nonnegative integer *k*, an array of point indices, nn_{idx}, and an array of distances, $dists$. Both arrays are assumed to contain at least *k* elements. This procedure computes the *k* nearest neighbors of *q* in the point set, and stores the indices of the nearest neighbors in the array nn_{idx}. Here, k is set equal to $knear$, in this article it is $k = 2$. In order to find point symmetry distance of a particular point $\overline{x}$ with respect to the centre $\overline{c}$, we have to find the first $knear$ nearest neighbors of $\overline{x}^*$ which is equal to $2 \times \overline{c} - \overline{x}$. Therefore the query point q is set equal to $\overline{x}^*$. After getting the $knear$ nearest neighbors of $\overline{x}^*$ the symmetrical distance of $\overline{x}$ with respect to a centre $\overline{c}$ is calculated using Equation 2.

B. The Cluster Validity Measure

1) Definition: The newly developed PS distance is used to define a cluster validity function which measures the overall average symmetry with respect to the cluster centers. This is inspired by the I-index developed in [10].

Consider a partition of the data set $X = \{\overline{x}_j : j = 1, 2, \ldots n\}$ into K clusters where the center of cluster $\overline{c}_i$ is computed by using $\overline{c}_i = \frac{\sum_{j=1}^{n_i} \overline{x}_j^i}{n_i}$ where n_i $(i = 1, 2, \ldots, K)$ is the number of points in cluster i and $\overline{x}_j^i$ denotes the jth point of the ith cluster. The new cluster validity function *Sym* is defined as:

$$Sym(K) = \left(\frac{1}{K} \times \frac{1}{\mathcal{E}_K} \times D_K\right). \quad (6)$$

Here,

$$\mathcal{E}_{\mathcal{K}} = \sum_{i=1}^{K} E_i, \quad (7)$$

such that

$$E_i = \sum_{j=1}^{n_i} d_{ps}^*(\overline{x}_j^i, \overline{c}_i) \quad (8)$$

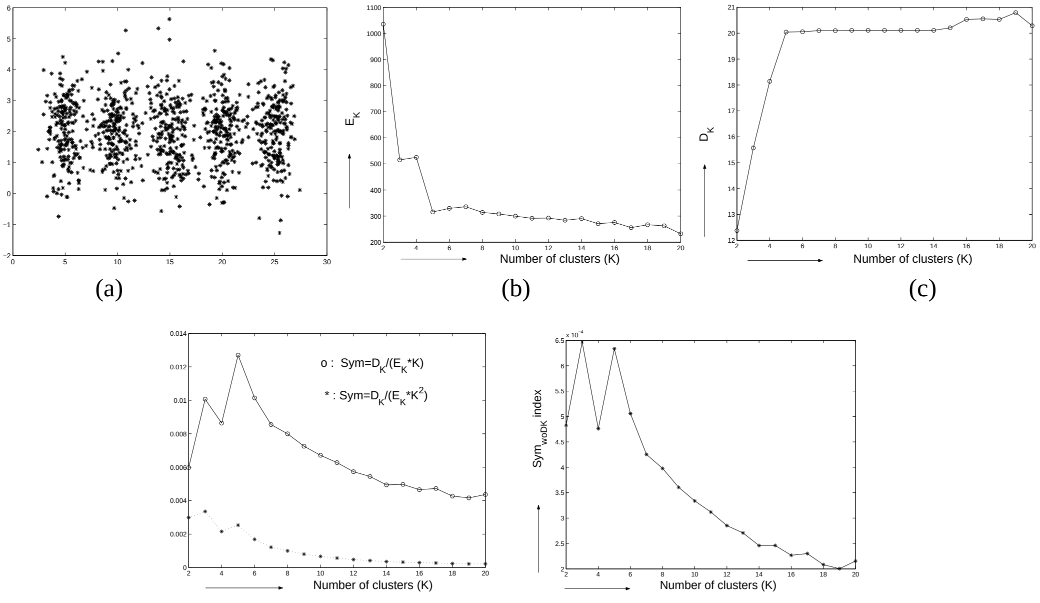
and

$$D_K = max_{i,j=1}^{K} \|\overline{c}_i - \overline{c}_j\| \quad (9)$$

D_K is the maximum Euclidean distance between two cluster centres among all pairs of centres. $d_{ps}^*(\overline{x}_j^i, \overline{c_i})$ is computed by Equation 2 with some constraint. Here, the first $knear$ nearest neighbors of $\overline{x}_j^* = 2 \times \overline{c}_i - \overline{x}_j^i$ will be searched among only those points which are in cluster i, i.e., the $knear$ nearest neighbors of $\overline{x}_j^*$, the reflected point of x_j^i with respect to $\overline{c}_i$, and x_j^i should belong to the ith cluster. The objective is to maximize this index in order to obtain the actual number of clusters.

2) Explanation: As formulated in Equation 6, *Sym*-index is a composition of three factors, $1/K$, $1/\mathcal{E}_{\mathcal{K}}$ and D_K. The first factor increases as K decreases; as *Sym*-index needs to be maximized for optimal clustering, this factor prefers to decrease the value of K. The second factor is a measure of the total within cluster symmetry. For clusters which have good symmetrical structures, $\mathcal{E}_{\mathcal{K}}$ value is less. Note that as K increases, in general, the clusters tend to become more symmetric. Moreover, as $d_e(\overline{x},\overline{c})$ in Equation 2 also decreases, $\mathcal{E}_{\mathcal{K}}$ decreases, resulting in an increase in the value of the *Sym*-index. Since *Sym*-index needs to be maximized, it will prefer to increase the value of K. Finally the third factor, D_K, measuring the maximum separation between a pair of clusters, increases with the value of K. Note that the value of D_K is bounded by the maximum separation between a pair of points in the data set. As these three factors are complementary in nature, so they are expected to compete and balance each other critically for determining the proper partitioning.

The use of D_K, as the measure of separation, requires further elaboration. Instead of using the maximum separation between two clusters, several other alternatives could have been used. For example, if D_K was the sum of pairwise inter cluster distances in a K-cluster structure, then it would increase largely with increase in the value of K. This might lead to the formation of maximum possible number of clusters equal to the number of elements in the data set. If D_K was the average inter cluster distance then it would decrease at each step with K, instead of being increased. So, this will only leave us with the minimum possible number of clusters. The minimum distance between two clusters may be another choice for D_K. However, this measure would also decrease significantly with increase in the number of clusters. So this would lead to a structure where the loosely connected substructures remain as they were, where in fact a separation was



(a) (b) (c)

(d) (e)

Fig. 2. (a) *Normal_2_5* (b) Variation of $\mathcal{E}_{\mathcal{K}}$ value with number of clusters (c) Variation of D_K value with number of clusters (d) Variation of two different forms of *Sym*-index with number of clusters (e) Variation of Sym_{woDk}-index ($Sym_{woDk} = \frac{1}{K \times \mathcal{E}_{\mathcal{K}}}$) with number of clusters

expected. Thus maximum separability may not be attained. In contrast, if we consider the maximum inter cluster separation then we see that this tends to increase significantly until we reach the maximum separation among compact clusters and then it becomes almost constant. The upper bound of this value, which is equal to the maximum separation between two points, is only attainable when we have two extreme data elements as two single element clusters. But the terminating condition is reached well before this situation. This is the reason why we try to improve the maximum distance between two maximally separated clusters.

C. The Interaction Between Different Components of Sym-index

In order to show how the different components of the newly proposed *Sym*-index compete with each other to determine the proper number of clusters from a data set, the variations of different components of *Sym*-index along with the number of clusters are shown pictorially for one artificially generated data set, *Normal_2_5*. This data set has 2000 two dimensional points distributed over 5 clusters. The variations of different components of the *Sym*-index along with the number of clusters are shown in Figure 2. Note that generally D_K increases rapidly with K upto a certain value of K, after which the rate of increase becomes less. In fact the upper bound of D_K is equal to the maximum separation between any two points in the data set. $\mathcal{E}_{\mathcal{K}}$ generally decreases with K. The overall variation of $\frac{D_K}{\mathcal{E}_{\mathcal{K}} \times K}$ versus K is plotted in Figure 2(d). We have also explored the variations of other forms of *Sym*-index with the number of clusters. Figure 2(d) also shows the variation of *Sym*-index, with K replaced by K^2 in the denominator, with the number of clusters. It shows that this form of *Sym*-index fails to identify the proper number of clusters from Normal_2_5 data. In order to establish the need to have D_K in *Sym*-index, Figure 2(e) shows the variation of the index (Sym_{woDK}-index) without having the factor D_K. Evidently the index then fails to identify the proper number of clusters.

III. VGAPS: Variable String Length Point Symmetry Based clustering technique

In this section a new clustering algorithm based on the proposed *Sym*-index and genetic algorithm is described in detail. It includes determination of the number of clusters as well as the appropriate clustering of the data set. This genetic clustering technique is subsequently referred to as variable string length genetic clustering technique with point symmetry based distance (VGAPS).

A. String Representation and Population Initialization

In VGAPS clustering, the chromosomes are made up of real numbers which represent the coordinates of the centers of the partitions. If chromosome i encodes the centers of K_i clusters in d dimensional space then its length l_i is taken to be $d * K_i$. For example, in three dimensional space, the chromosome $< 12.3\ 1.4\ 5.6 \quad 22.1\ 0.01\ 10.2 \quad 0.0\ 5.3\ 15.3 \quad 13.2\ 10.2\ 7.5 >$ encodes 4 cluster centers, (12.3, 1.4, 5.6), (22.1, 0.01, 10.2), (0.0, 5.3, 15.3) and (13.2, 10.2, 7.5). Each center is considered to be indivisible. Each string i in the population initially encodes the centers of a number, K_i, of clusters, such that $K_i = (rand()\text{mod}(K_{max}-1))+2$. Here, $rand()$ is a function returning an integer, and K_{max} is a soft estimate of the upper bound of the number of clusters. The number of clusters will therefore range from two to K_{max}. The K_i centers encoded in a chromosome are randomly selected distinct points from the data set.

B. Fitness Computation

Fitness computation is composed of two steps. Firstly points are assigned to different clusters using the point symmetry based distance, d_{ps}. Next, the cluster validity index, *Sym*-index, is computed and used as a measure of the fitness of the chromosome.

1) Assignment of points: Here each point $\overline{x}_i$, $1 \leq i \leq n$ is assigned to cluster k iff $d_{ps}(\overline{x}_i, \overline{c}_k) \leq d_{ps}(\overline{x}_i, \overline{c}_j)$, $j = 1, \ldots, K$, $j \neq k$ and $d_{sym}(\overline{x}_i, \overline{c}_k) \leq \theta$. Here θ is a threshold described later. For $d_{sym}(\overline{x}_i, \overline{c}_k) > \theta$, point $\overline{x}_i$ is assigned to some cluster m iff $d_e(\overline{x}_i, \overline{c}_m) \leq d_e(\overline{x}_i, \overline{c}_j)$, $j = 1, 2 \ldots K$, $j \neq m$. In other words, point $\overline{x}_i$ is assigned to that cluster with respect to whose center its PS-distance is the minimum, provided this value is less than some threshold θ. Otherwise assignment is done based on the minimum Euclidean distance criterion as normally used in [16] or the *K*-means algorithm. The reason for doing such an assignment is as follows: In the intermediate stages of the algorithm, when the centers are not yet properly evolved, then the minimum d_{ps} value for a point is expected to be quite large, since the point might not be symmetric with respect to any center. In such cases, using Euclidean distance for cluster assignment appears to be intuitively more appropriate.

We also provide a rough guideline of the choice of θ, the threshold value on the PS-distance. It is to be noted that if a point is indeed symmetric with respect to some cluster center then the symmetrical distance computed in the above way will be small, and can be bounded as follows. Let d_{NN}^{max} be the maximum nearest neighbor distance in the data set. That is $d_{NN}^{max} = \max_{i=1,\ldots N} d_{NN}(\overline{x}_i)$, where $d_{NN}(\overline{x}_i)$ is the nearest neighbor distance of $\overline{x}_i$. Let us assume that the reflected point of $\overline{x}$ with respect to the cluster center $\overline{c}$ lies near any point in the data space. Ideally, a point $\overline{x}$ is exactly symmetrical with respect to some $\overline{c}$ if $d_1 = 0$. However considering the uncertainty of the location of a point as a sphere of radius $d_{NN}^{max}/2$ around it, we can bound d_1 as $d_1 \leq \frac{d_{NN}^{max}}{2}$ and $d_2 \leq \frac{3 \times d_{NN}^{max}}{2}$, resulting in

$$\frac{d_1 + d_2}{2} \leq d_{NN}^{max}$$

Thus, we have kept the threshold θ equals to d_{NN}^{max}, making its computation automatic and without user intervention. After the assignments are done, the cluster centers encoded in the chromosome are replaced by the mean points of the respective clusters. This is referred to as the *k*-means like update center operation.

2) Fitness Calculation: The fitness of a chromosome is computed using the newly developed *Sym*-index. The objective is to maximize this index in order to obtain the actual number of clusters and to achieve proper clustering. The fitness function for chromosome j is defined as Sym_j, where Sym_j is the Sym-index computed for the chromosome. This fitness function is maximized using a genetic algorithm.

C. Genetic Operations and Terminating Criterion

The following genetic operations are performed on the population of strings for a number of generations.

1) Selection: The selection operator randomly selects a chromosome from the previous population according to the distribution given by

$$P(s_i) = \frac{F(s_i)}{\sum_{j=1}^{N} F(s_j)} \tag{10}$$

where $F(s_i)$ represents fitness value (Sym-index) of the string s_i in the population and N denotes the population size. Here, a string receives a number of copies that is proportional to its fitness in the population.

2) Crossover: For the purpose of crossover, the cluster centers are considered to be indivisible, i.e., the crossover points can only lie in between two cluster centers. The crossover operation, applied stochastically, must ensure that information exchange takes place in such a way that both the offspring encode the centers of at least two clusters. For this purpose, the operator is defined as follows [24]: Let parent chromosomes P_1 and P_2 encode M_1 and M_2 cluster centers, respectively. The crossover point, τ_1, in P_1 is generated as τ_1=rand() mod M_1. Let τ_2 be the crossover point in P_2; it may vary in between [LB(τ_2),UB(τ_2)], where LB(τ_2) and UB(τ_2) indicate the lower and upper bounds of the range of τ_2, respectively. LB(τ_2) and UB(τ_2) are given by $\text{LB}(\tau_2) = \min[2, \max[0, 2 - (M_1 - \tau_1)]]$ and $\text{UB}(\tau_2) = [M_2 - \max[0, 2 - \tau_1]]$. Therefore τ_2 is given by

$$\begin{aligned} \tau_2 &= \text{LB}(\tau_2) + \text{rand}()\text{mod}(\text{UB}(\tau_2) - \text{LB}(\tau_2)), \text{if}(UB(\tau_2) \geq LB(\tau_2)), \\ \tau_2 &= 0 \text{ otherwise.} \end{aligned}$$

It can be verified by some simple calculations that if the crossover points τ_1 and τ_2 are chosen according to the above rules, then none of the offspring generated would have less than two clusters.

Crossover probability, p_c, is selected adaptively as in [25]. The expressions for crossover probabilities are given below:

$$\begin{aligned} p_c &= k_1 \times \frac{(f_{max} - f')}{(f_{max} - \overline{f})}, \quad \text{when} \quad f' \geq \overline{f}, && (11) \\ &= k_3, \quad \text{when} \quad f' < \overline{f}, && (12) \end{aligned}$$

where f_{max} is the maximum fitness value of the current population, $\overline{f}$ is the average fitness value of the population and f' is the larger of the fitness values of the solutions to be crossed. Here the values of k_1 and k_3 are kept equal to 1.0 [25]. The aim behind such adaptation is to achieve a trade-off between exploration and exploitation in a different manner by varying probability of crossover, p_c, and probability of mutation, p_m, adaptively in response to the fitness values of the solutions; p_c and p_m are increased when the population tends to get stuck at a local optimum and are decreased when the population is scattered in the solution space.

3) Mutation: Three types of mutations are considered here.

- A valid position (i.e., which is not '#') is replaced with a random variable drawn from a Laplacian distribution, $p(\epsilon) \propto e^{-\frac{|\epsilon - \mu|}{\delta}}$, where the scaling factor δ sets the magnitude of perturbation. Here μ is the value at the position which is to be perturbed. The scaling factor δ is chosen equal to 2. The old value at the position is replaced with the newly generated value.
- A randomly chosen valid position is removed and replaced by '#'.
- A randomly chosen invalid position is replaced by randomly chosen point from the data set.

Any one of the above mentioned types of mutation is applied on each chromosome of the population with some probability of mutation, p_m. The mutation probability is selected adaptively for each chromosome as in [25]. The expression for mutation probability, p_m, is given below:

$$\begin{aligned} p_m &= k_2 \times \frac{(f_{max} - f)}{(f_{max} - \overline{f})} \quad \text{where } f \geq \overline{f}, && (13) \\ &= k_4 \quad \text{where} \quad f < \overline{f}. && (14) \end{aligned}$$

Here, values of k_2 and k_4 are kept equal to 0.5.

4) Termination Criterion: In this article the processes of fitness computation, selection, crossover, and mutation are executed for a maximum number of generations. The best string having the largest fitness (i.e., the largest *Sym*-index value) seen up to the last generation provides the solution to the clustering problem. We have implemented elitism at each generation by preserving the best string seen up to that generation in a location outside the population and also inside the population by replacing the string with lowest fitness value. Thus on termination, this location contains the centers of the final clusters.

IV. On The Convergence Property of VGAPS

It has been shown using the finite Markov chain theory that the canonical genetic algorithms converge to the global optimum [26]. In [27] it is also been proved along the lines of [26] that Genetic K-means algorithm also converges to the global optimum depending on some conditions on its parameters. Here the global convergence of VGAPS will be proved along the similar lines by deriving some conditions on the parameters of VGAPS that ensures the global convergence.

Consider the process $\{\mathcal{P}(t)\},\ t \geq 0$, where $\mathcal{P}(t)$ represents the population maintained by VGAPS at generation t. The state space of this process is the space of all possible populations $\mathcal{S}$; and the states are numbered from 1 to $|\mathcal{S}|$. Here the state space comprises the populations containing strings representing partitions with K clusters where $K \in [K_{min}, K_{max}]$. From the definition of VGAPS, $\mathcal{P}(t+1)$ can be determined completely by $\mathcal{P}(t)$, i.e.,

$$\begin{aligned} & Pr\{\mathcal{P}(t)=p_t|\mathcal{P}(t-1)=p_{t-1},\ldots,\mathcal{P}(0)=p_0\} \\ = \quad & Pr\{\mathcal{P}(t)=p_t|\mathcal{P}(t-1)=p_{t-1}\}. \end{aligned}$$

Hence $\{\mathcal{P}(t)\},\ t\ \geq 0$, is a Markov chain. Also, the transition probabilities are independent of the time instant, i.e., if

$$p_{ij}(t) = Pr\{\mathcal{P}(t)=p_j|\mathcal{P}(t-1)=p_i\}$$

then $p_{ij}(s) = p_{ij}(t)$ for all $p_i, p_j \in \mathcal{S}$ and for all $s, t \geq 1$. Therefore, $\{\mathcal{P}(t)\},\ t \geq 0$ is a time-homogeneous finite Markov chain. Let $\mathbf{P} = (p_{ij})$ be the *transition matrix* of the process $\{\mathcal{P}(t)\},\ t \geq 0$. The entries of the matrix $\mathbf{P}$ satisfy $p_{ij} \in [0,1]$ and $\sum_{j=1}^{|\mathcal{S}|} p_{ij} = 1,\ \forall i \in \mathcal{S}$. Any matrix whose entries satisfy the above conditions is called a *stochastic matrix*. Some definitions are given below which will be used in the rest of this section.

A square matrix $\mathbf{A}_{m\times m}$ is said to be positive, if $a_{ij} > 0, \forall i,j \in \{1,2,\ldots,m\}$ and is said to be *primitive*, if there exists a positive integer k such that $\mathbf{A}^k$ is positive. A square matrix is said to be *column-allowable*, if it has at least one positive entry in each column.

In the following theorem it is required that $\mathbf{P}$ be a primitive matrix. So, first we investigate the conditions on the operators which make the matrix $\mathbf{P}$ primitive. The probabilistic changes of the chromosome within the population caused by the operators used in VGAPS are captured by the transition matrix $\mathbf{P}$, which can be decomposed in a natural way into a product of stochastic matrices

$$\mathbf{P} = \mathbf{K} \times \mathbf{C} \times \mathbf{M} \times \mathbf{S}, \tag{15}$$

where $\mathbf{K}$, $\mathbf{C}$, $\mathbf{M}$ and $\mathbf{S}$ describe the intermediate transitions caused by K-means like update center operator, crossover operator, mutation and selection operators, respectively. It is easy to consider that all these matrices are stochastic matrices.

Proposition 1. *Stochastic matrices form a group under matrix multiplication.*

Thus for the two stochastic matrices, $\mathbf{K}$ and $\mathbf{C}$, by proposition 1, $\mathbf{C}^{'} = \mathbf{K} \times \mathbf{C}$ is also a stochastic matrix. Therefore Equation 15 can be written as

$$\mathbf{P} = \mathbf{C}^{'} \times \mathbf{M} \times \mathbf{S}, \tag{16}$$

where $\mathbf{C}^{'}$, $\mathbf{M}$ and $\mathbf{S}$ are stochastic matrices.

Proposition 2. *Let $\mathbf{C}^{'}$, $\mathbf{M}$ and $\mathbf{S}$ be stochastic matrices, where $\mathbf{M}$ is positive and $\mathbf{S}$ is column-allowable. Then the product $\mathbf{C}^{'} \times \mathbf{M} \times \mathbf{S}$ is positive.*

Since every positive matrix is primitive, it is therefore, enough to find the conditions which make $\mathbf{M}$ positive and $\mathbf{S}$ column-allowable.

A. To check whether the mutation matrix is positive

The matrix $\mathbf{M}$ is positive if any string $s \in \mathcal{S}$ can be obtained from any other string on application of the corresponding mutation operator. The mutation operator defined in the earlier section ensures the above condition. The mutation operator is of three types. The first type is for obtaining a valid position from any other valid position. By generating a random variable using a Laplacian distribution, there is a non-zero probability of generating any valid position from any other valid position while probability of generating a value near the old value is more. The second type is for decreasing the value of K i.e., from a chromosome consisting of K_1 number of centers, another chromosome of having K_2 number of clusters, where $K_1 > K_2$, is generated by this type of mutation operation. The third type of mutation operator is for increasing the value of K in a particular chromosome, i.e., if a chromosome encodes K_1 clusters, where $K_1 < K_{max}$, then by the third type of mutation operation some new cluster centers can be included in it, increasing in number of clusters.

The above discussion implies that the mutation operation can change any string to any other string in the search space with nonzero probability. Hence, the transition matrix, $\mathbf{M}$, corresponding to the above mutation operator is positive.

B. Conditions on Selection

The probability of survival of a string in the current population depends on the fitness value of the string; so is the transition matrix due to selection, $\mathbf{S}$. Very little can be said about $\mathbf{S}$ if the fitness function is defined as only the *Sym*-index value of that particular partition. The following modification to the fitness function will ensure the column allowability of S. Let

$$F(s) = c_s \times Sym_{max} + Sym(s) \tag{17}$$

where Sym_{max} is the maximum *Sym*-index value that has been encountered till the present generation and $c_s \geq 1$. *Sym(s)* is the *Sym*-index value of the sth string. Then the fitness value of each chromosome in the population is strictly positive. Therefore, the probability that selection does not alter the present state, s_{ii} can be bounded as follows:

$$s_{ii} \geq \frac{F(s_1)}{\sum_{l=1}^{N} F(s_l)} \times \frac{F(s_2)}{\sum_{l=1}^{N} F(s_l)} \ldots \times \frac{F(s_N)}{\sum_{l=1}^{N} F(s_l)}$$

$$= \frac{\prod_{l=1}^{N} F(s_l)}{(\sum_{l=1}^{N} F(s_l))^N} > 0 \forall i \in \mathcal{S}$$

where s_l is the lth string of the current population. Even though this bound changes with the generation, it is always strictly positive; hence selection matrix $\mathbf{S}$ is *column-allowable*.

Theorem IV.1. *Let $X(t) = Sym(s^*(t))$, where $s^*(t)$ is the string with maximum Sym-index value, encountered during the evolution of VGAPS till the time instant t. Let the mutation operator be as same as defined in subsection III-C.3, and the fitness function be as defined in Equation 17. Then*

$$\lim_{t \to \infty} Pr\{X(t) = \mathit{Sym}^*\} = 1 \tag{18}$$

where Sym* = max$\{Sym(i) | i \in \mathcal{T}\}$, $\mathcal{T}$ is the set of all legal strings.

Proof: It is proved (Ref. [26], Theorem 6) that a canonical GA whose transition matrix is primitive and which maintains the best solution found over time, converges to the global optimum in the sense given in Equation 18. As proved in Proposition 2, the transition matrix of VGAPS is primitive. Moreover, VGAPS uses elitist model of GA, i.e., it preserves the best solution found till the current time instant. Thus, the above theorem follows from ([Ref. [26], Theorem 6]).

The above theorem implies that $X(t)$, the maximum *Sym*-index value of the strings encountered by VGAPS till the instant t, converges to the global optimum *Sym**, with probability 1.

V. Data Sets Used and Implementation Results

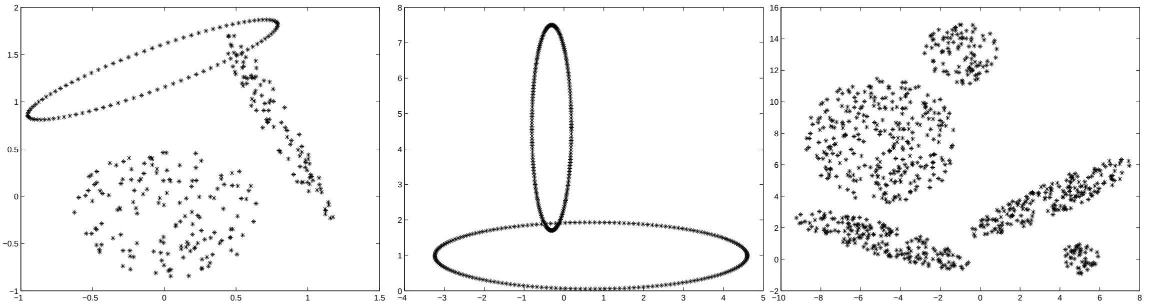
This section provides a description of the data sets and the implementation results of the proposed algorithm. Nine artificial and five real life data sets are used for the experiments.

A. Data Sets Used

1) Artificial data sets:
   a) *Data1*: This data set, used in [3], is a combination of ring-shaped, spherically compact and linear clusters shown in Figure 3(a). The total number of points in it is 350.
   b) *Data2*: This data set contains 400 points distributed on two crossed ellipsoidal shells shown in Figure 3(b).
   c) *Data3*: This data set contains 850 data points distributed on five clusters, as shown in Figure 3(c).
   d) *Data4*: This data set, used in [16], consists of 250 two dimensional data points distributed over 5 spherically shaped clusters. The clusters present in this data set are highly overlapping, each consisting of 50 data points. This data set is shown in Figure 4(a). The clusters present in this data set are symmetrical, some of which are symmetrical interclusters.
   e) *Data5*: This data set, used in [16], consists of 300 data points distributed over 6 different clusters in two dimensions. The clusters are of same sizes. This data set is shown in Figure 4(b). There are some symmetrical interclusters in this data set.
   f) *Data6*: This is a two-dimensional data set, used in [28], composed of three clusters shown in Figure 4(c). This data set consists of two small clusters (one has six elements and the other has three) separated by a large (40 element) cluster.
   g) *Data7*: This data set contains 398 points distributed on two non-overlapping ellipsoidal shells in three dimensions as shown in Figure 5(a).
   h) *Data8*: This data set contains 598 points distributed on two non-overlapping ellipsoidal shells and in one elliptical shaped cluster in three dimensions as shown in Figure 5(b).
   i) *Pat*: This data set, used in [29], consists of 2 non-linear, non-overlapping and asymmetric clusters. The data set is shown in Figure 5(c).
2) Real-life data sets: The 5 real life data sets were obtained from [30].
   a) *Iris*: *Iris* data set consists of 150 data points distributed over 3 clusters. Each cluster consists of 50 points. This data set represents different categories of irises characterized by four feature values [31]. It has three classes Setosa, Versicolor and Virginica. It is known that two classes (Versicolor and Virginica) have a large amount of overlap while the class Setosa is linearly separable from the other two. A two dimensional projection of the data set is shown in Figure 6(a).
   b) *Breast Cancer*: This *Wisconsin Breast Cancer* data set consists of 683 sample points. Each pattern has nine features corresponding to clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli and mitoses. There are two categories in the data: malignant and benign. The two classes are known to be linearly separable.
   c) *Newthyroid*: The original database from where it has been collected is titled as Thyroid gland data ('normal', 'hypo' and 'hyper' functioning). Five laboratory tests are used to predict whether a patient's thyroid belongs to the class euthyroidism, hypothyroidism or hyperthyroidism. There are a total of 215 instances and the number of attributes is five. A two dimensional projection of the data set is shown in Figure 6(b).
   d) *Glass*: This is a glass identification data consisting of 214 instances having 9 features (an Id# feature has been removed). The study of the classification of the types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence, if it is correctly identified. There are 6 categories present in this data set. A two dimensional projection of the data set is shown in Figure 6(c).
   e) *LiverDisorder*: This is the Liver Disorder data consisting of 345 instances having 6 features each. The data has two categories.

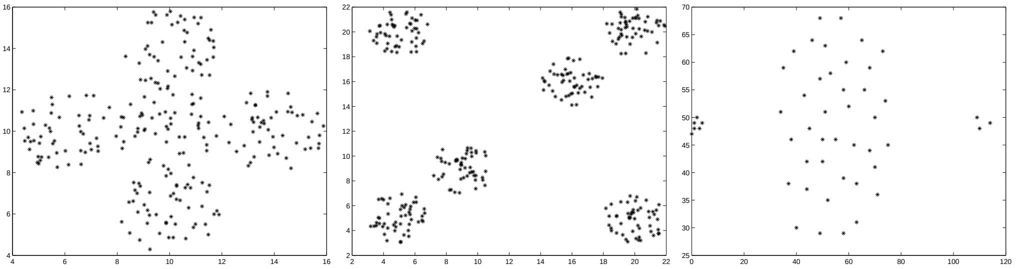
B. Results and Discussions

In VGAPS-clustering, the population size P is taken to be equal to 100. K_{min} and K_{max} are set equal to 2 and $\sqrt{n}$, respectively, where n is the total number of data points in the particular data set. VGAPS is executed for a total of 30 generations. Note that it is shown in Ref. [32] that if exhaustive enumeration is used to solve a clustering problem with *n* points and *K* clusters, then one requires to evaluate $1/K \sum_{j=1}^{K} (-1)^{K-j} j^n$ partitions. For a data set of size 10 with 2 clusters, this value is $2^9 - 1 (= 511)$, while



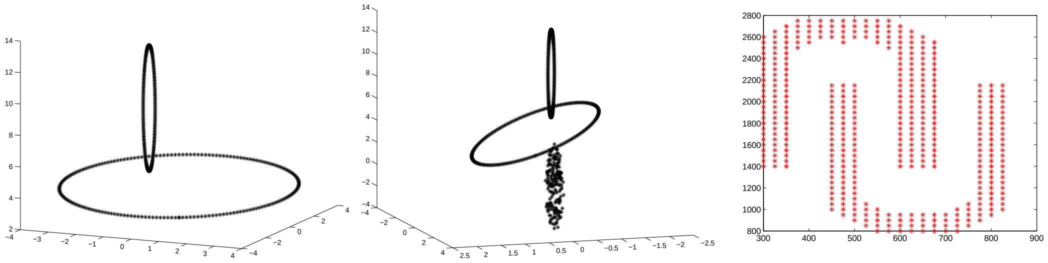
(a) (b) (c)

Fig. 3. (a) *Data1* dataset (b) *Data2* dataset (c) *Data3* dataset



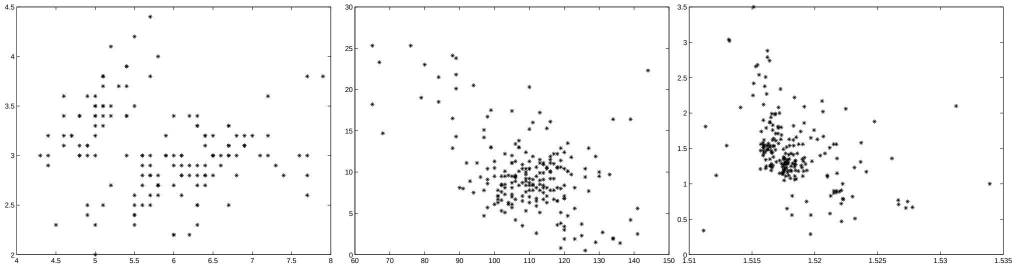
(a) (b) (c)

Fig. 4. (a) *Data4* dataset (b) *Data5* dataset (c) *Data6* dataset



(a) (b) (c)

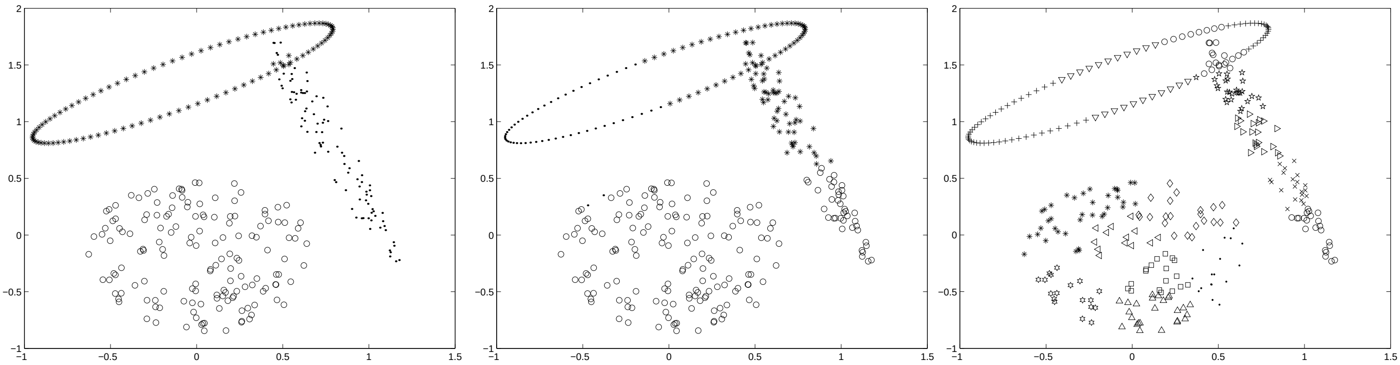
Fig. 5. (a) *Data7* dataset (b) *Data8* dataset (c) *Pat* dataset



(a) (b) (c)

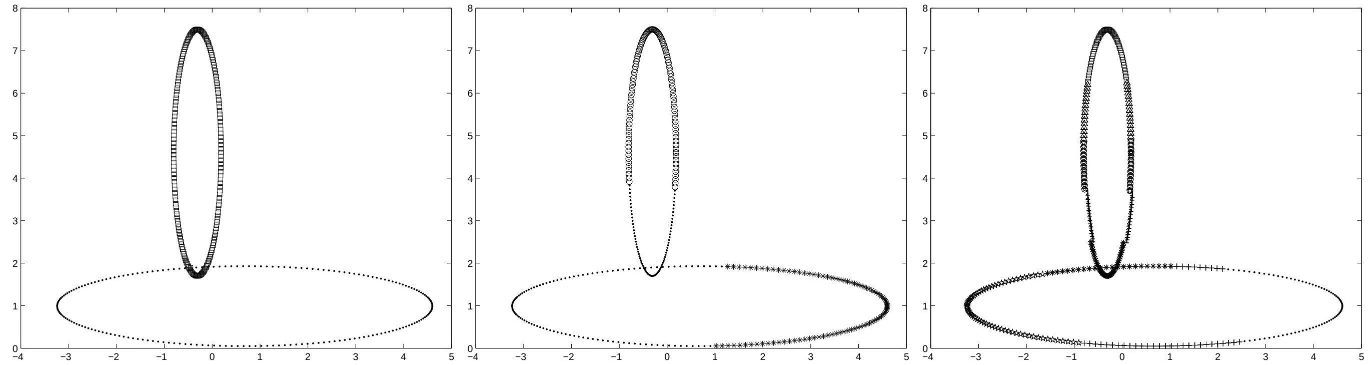
Fig. 6. (a) *Iris* data set projected on the first two dimensions (b) *Newthyroid* data set projected on the first two dimensions (c) *Glass* data set projected on the first and the fourth dimensions

that of size 50 with 2 clusters, it is $2^{49} - 1$ (i.e., of the order of 10^{15}). If the number of clusters is not specified a priori, then the search space will become even larger and the effectiveness of GAs becomes more evident. In order to evaluate the proposed method, we performed two types of experiments. At first we show that VGAPS optimizing *Sym*-index performs better than VGAPS optimizing two other indices, viz., PS-index [33] and $\mathcal{I}$-index [10]. After that, we explore the properties of the VGAPS



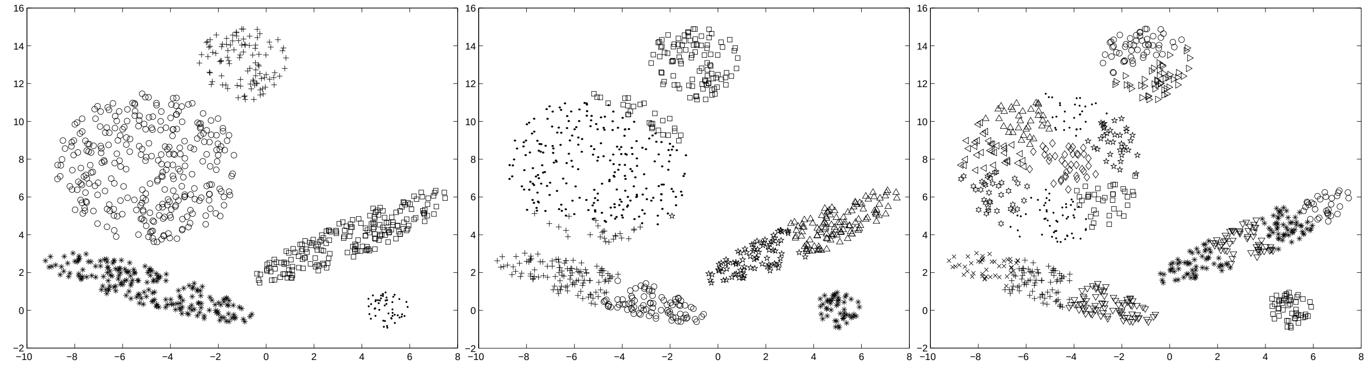
(a) (b) (c)

Fig. 7. Clustered *Data1* after application of (a) VGAPS-clustering where 3 clusters are detected (b) GCUK-clustering where 3 clusters are detected (c) HNGA-clustering where 16 clusters are detected



(a) (b) (c)

Fig. 8. Clustered *Data2* after application of (a) VGAPS-clustering where 2 clusters are detected (b) GCUK-clustering where 3 clusters are detected (c) HNGA-clustering where 8 clusters are detected



(a) (b) (c)

Fig. 9. Clustered *Data3* using (a) VGAPS-clustering where 5 clusters are detected (b) GCUK-clustering where 7 clusters are detected (c) HNGA-clustering where 18 clusters are detected

optimizing *Sym*-index and compare its performance with other genetic clustering methods, which do not need knowledge about the number of clusters *a priori*.

1) Exploring Sym-index as fitness function: In the first experiment, we establish the effectiveness of using the *Sym*-index with VGAPS-clustering vis-a-vis another point symmetry based validity index, PS-index [33] and a Euclidean distance based cluster validity index, $\mathcal{I}$-index [10]. The number of clusters obtained after applying VGAPS optimizing these three validity indices separately for all the data sets are shown in Table I. It can be seen from the table that VGAPS-clustering with *Sym*-index is able to find out the proper cluster number from data sets having symmetrical shaped clusters. VGAPS-clustering with $\mathcal{I}$-index is able to find the proper cluster number from data sets with spherically symmetrical structure but it is not able to detect other shaped clusters. It is because the $\mathcal{I}$-index essentially prefers hyperspherical clusters, which is not the case for *Data1*, *Data2*, *Data7*, *Data8*, and *Pat*. VGAPS-clustering with PS-index is able

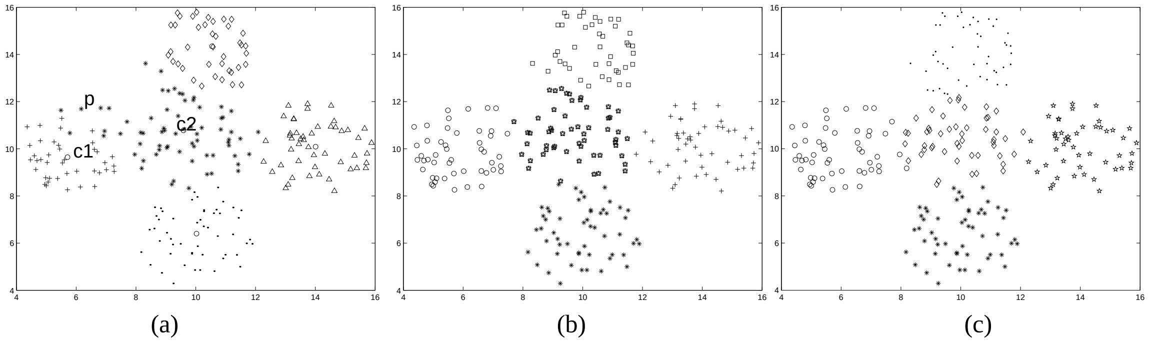


Fig. 10. Clustered *Data4* using (a) VGAPS-clustering where 5 clusters are detected (b) GCUK-clustering where 5 clusters are detected (c) HNGA-clustering where 5 clusters are detected

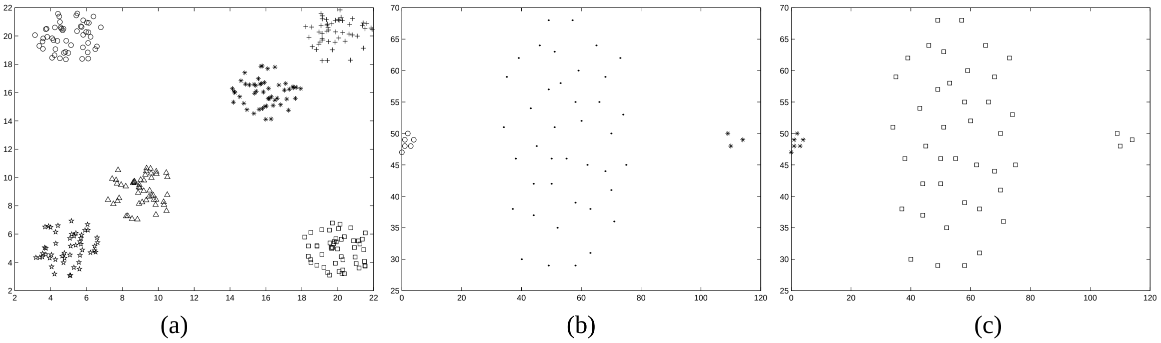


Fig. 11. (a) Clustered *Data5* by VGAPS-clustering, GCUK-clustering and HNGA-clustering where 6 clusters are detected (b) Clustered *Data6* by VGAPS-clustering and HNGA-clusetring where 3 clusters are detected (c) Clustered *Data6* by GCUK-clustering where 2 clusters are detected

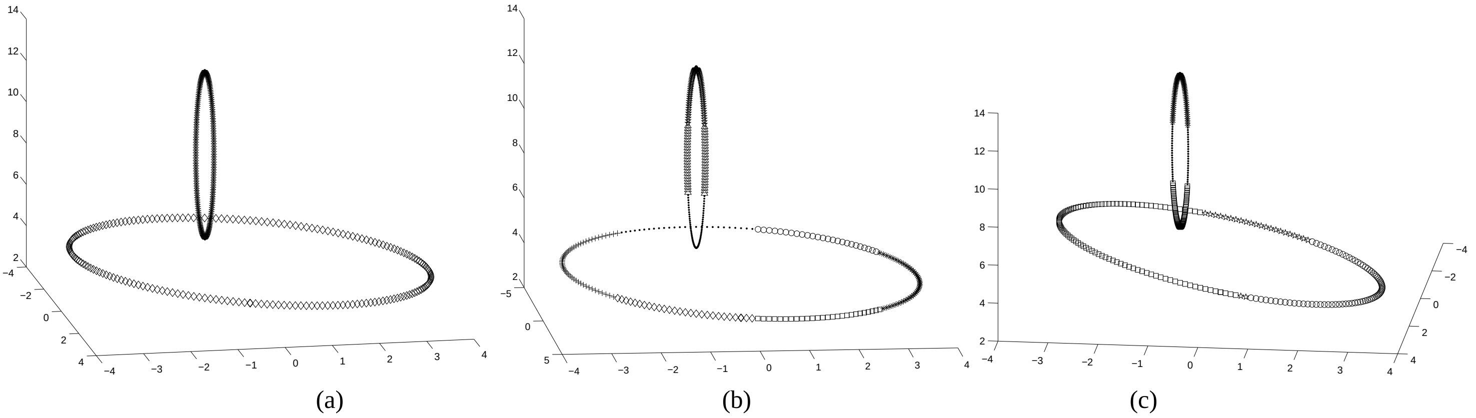
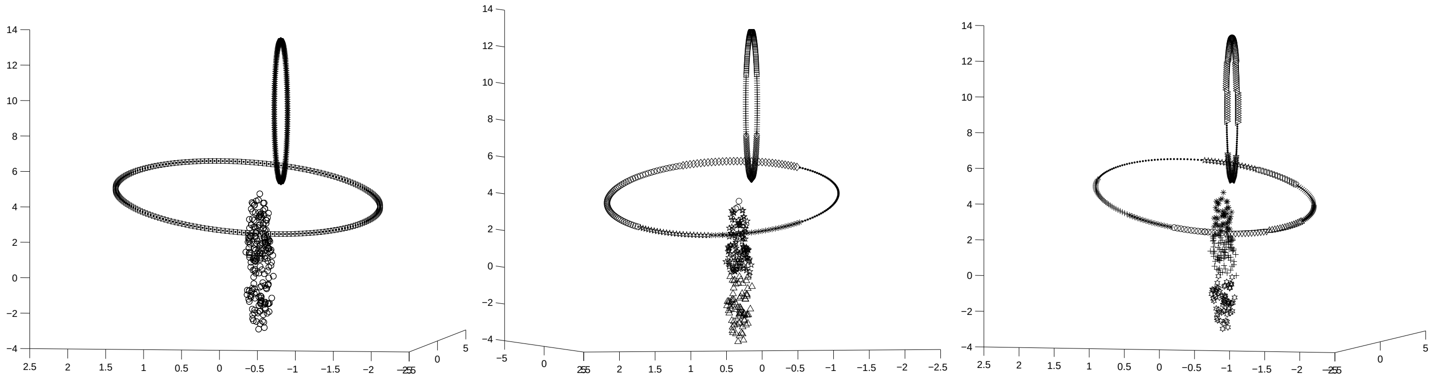


Fig. 12. Clustered *Data7* using (a) VGAPS-clustering where 2 clusters are detected (b) GCUK-clustering where 8 clusters are detected (c) HNGA-clustering where 5 clusters are detected

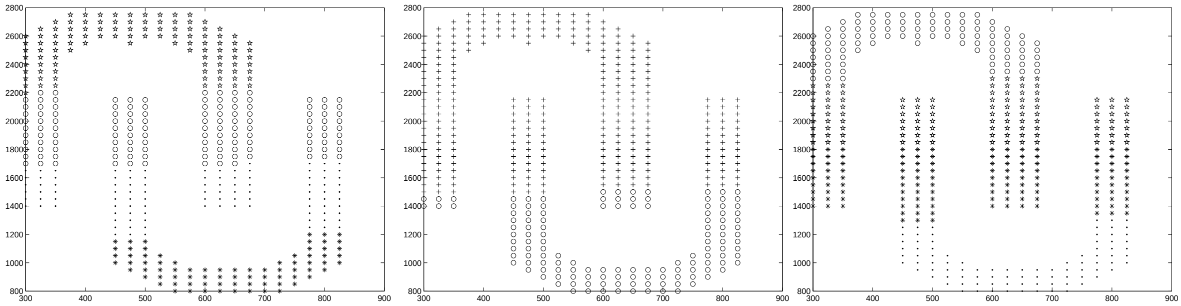
to detect the proper clusters from those data sets where the clusters have strong point symmetry. However, as discussed in [3], the definition of point symmetry distance in PS-index precludes the detection of symmetrical interclusters. Thus it fails for *Data4* and *Data5* which have clearly symmetrical interclusters.

2) Exploring the VGAPS-clustering: In this section, we compare the performance of the VGAPS-clustering with those of the GCUK-clustering [16] and a recently developed HNGA clustering [18]. Before discussing the comparative experiments, we first briefly describe the GCUK-clustering and the HNGA-clustering methods. In the GCUK-clustering, variable string length Genetic Algorithm (VGA) [17] was applied with real parameter representation as the underlying search tool. The chromosome encodes the centers of a number of clusters, whose value may vary. Modified versions of crossover and mutation are used. Davies-Bouldin cluster validity index is utilized for computing the fitness of the chromosomes. In Hybrid Niching Genetic Algorithm (HNGA) [18], a weighted sum validity function (WSVF), which is a weighted sum of several normalized cluster validity functions, is used for optimization to automatically evolve the proper number of clusters and the appropriate partitioning of the data set. Within the HNGA, a niching method is developed to prevent premature convergence during the search. Additionally, in order to improve the computational efficiency, a hybridization between the niching



(a) (b) (c)

Fig. 13. Clustered *Data8* using (a) VGAPS-clustering where 3 clusters are detected (b) GCUK-clustering where 8 clusters are detected (c) HNGA-clustering where 17 clusters are detected



(a) (b) (c)

Fig. 14. Clustered *Pat* using (a) VGAPS-clustering where 4 clusters are detected (b) GCUK-clustering where 2 clusters are detected (c) HNGA-clustering where 4 clusters are detected

method with the computationally attractive *K*-means is made. Here WSVF is defined as $WSVF = \sum_{i=1}^{m} w_i f_i(x)$ where m is the number of component functions, specifically $m = 6$ is used here. w_is are the non-negative weighting coefficients representing the relative importance of the functions such that $\sum_{i=1}^{m} w_i = 1$, and $f_i(x)$ are component functions (as used in [18]) corresponding to 1/(DB-index [34]), SIL-index [35], Dunn-index [36], Generalized Dunn-index [37], CH-index [38] and $\mathcal{I}$-index [10], respectively. Here weighting coefficients are chosen as $w_1 = w_2 = \ldots = w_m = 1/m$.

Table I shows the number of clusters identified by the three clustering algorithms for all the data sets. As is evident from Table I, except for *Pat* and *LiverDisorder*, for the other data sets VGAPS is able to find out the appropriate number of clusters and the proper partitioning for all the data sets. Figures 7(a), 8(a), 9(a), 10(a), 11(a), 11(b), 12(a), 13(a) and 14(a) show the final segmentation obtained after application of VGAPS on *Data1*, *Data2*, *Data3*, *Data4*, *Data5*, *Data6*, *Data7*, *Data8*, and *Pat*, respectively. Although for *Data4*, VGAPS is able to detect the clusters reasonably well, it is found to somewhat over-approximate the central cluster (which extends to the left). The reason is as follows. Let us take a point p which actually belongs to the left cluster but after application of VGAPS it is included in the central cluster (shown in Figure 10(a)). It can be seen from the figure that even though $d_e(p, c2)$ is slightly greater than $d_e(p, c1)$ but since $d_{sym}(p, c2)$ is significantly smaller than $d_{sym}(p, c1)$, p is assigned to the central cluster. As expected for dataset *Pat*, VGAPS is not able to detect the proper partitioning since the clusters are not symmetrical. For real-life data sets, it is not possible to show the segmentation results visually as these are higher dimensional data sets. Here segmentation results of three of these multi-dimensional data sets, just for an illustration, are provided by projecting the clustered data in some two dimensional feature space. Figures 15(a), 16(a) show, respectively, the segmentation results projected on first two feature space obtained on *Iris* and *Newthyroid* by VGAPS-clustering technique. Figure 17(a) shows the segmentation result by VGAPS-clustering on *Glass* data set projected on the first and the fourth feature space. The results on real-life data sets are therefore quantitatively compared with respect to the Minkowski scores described later. In order to validate the clustering results, the Hinton diagrams representing the similarities of all pairs of actual and observed cluster centers according to the Euclidean distance for three real-life data sets, *Iris*, *Newthyroid* and *Glass*, are shown in Figures 18-20 just for illustration. Similar diagrams can be also obtained for other two real-life data sets (these are not provided here due to restriction in page limit). Here hinton diagrams are sorted by columns so that the minimum squares corresponding to each row are along the diagonal. It is clearly evident from these diagrams that the cluster centers provided by VGAPS-clustering are very close to actual cluster centers.

Final clustering results obtained after the application of GCUK algorithm on the 9 artificial data sets are also shown in Figures 7(b), 8(b), 9(b), 10(b), 11(a), 11(c), 12(b), 13(b) and 14(b), respectively. Results shown in Table I reveals that GCUK-clustering

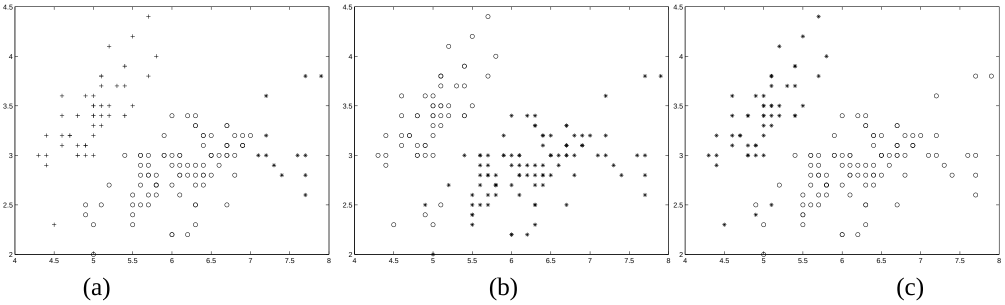


Fig. 15. Clustered *Iris* data projected on the first two dimensions using (a) VGAPS-clustering where 3 clusters are detected (b) GCUK-clustering where 2 clusters are detected (c) HNGA-clustering where 2 clusters are detected

TABLE I

COMPARING THE NUMBER OF CLUSTERS FOUND ON THE EXPERIMENTAL DATA SETS BY VGAPS-CLUSTERING USING *Sym*-INDEX, PS-INDEX AND $\mathcal{I}$-INDEX AS THE CLUSTER OBJECTIVE FUNCTION FOR COMPUTING FITNESS, GCUK-CLUSTERING AND HNGA-CLUSTERING. HERE AC DENOTES ACTUAL NUMBER OF CLUSTERS PRESENT IN THE DATA AND OC DENOTES THE OBTAINED NUMBER OF CLUSTERS.

Data Set	AC	OC by VGAPS using			OC by different methods		
		Sym	$\mathcal{I}$	PS	VGAPS	GCUK	HNGA
Data1	3	3	8	3	3	3	16
Data2	2	2	8	2	2	3	8
Data3	5	5	5	5	5	7	18
Data4	5	5	6	4	5	5	5
Data5	6	6	6	4	6	6	6
Data6	3	3	3	3	3	2	3
Data7	2	2	3	9	2	8	5
Data8	3	3	4	9	3	8	17
Pat	2	4	5	3	4	2	4
Iris	3	3	3	2	3	2	2
Cancer	2	2	2	2	2	2	2
Newthy roid	3	3	7	8	3	8	5
Glass	6	6	4	2	6	3	6
Liver Disorder	2	3	5	3	3	2	2

TABLE II

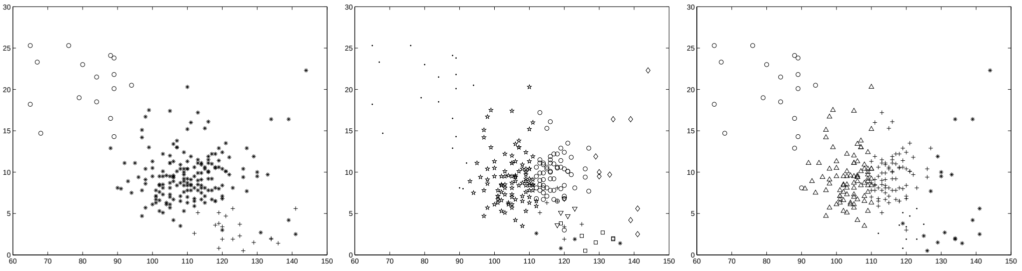
MINKOWSKI SCORES OBTAINED BY THREE ALGORITHMS FOR ALL DATA SETS USED HERE FOR EXPERIMENT

Data Set	VGAPS-clustering	GCUK-clustering	HNGA-clustering
Data1	0.12 ± 0.00	1.05 ± 0.02	0.85 ± 0.02
Data2	0.00 ± 0.00	0.98 ± 0.01	0.89 ± 0.012
Data3	0.00 ± 0.00	1.12 ± 0.021	0.9 ± 0.001
Data4	0.25 ± 0.02	0.14 ± 0.001	0.10 ± 0.002
Data5	0 ± 0.00	0 ± 0.00	0 ± 0.00
Data6	0 ± 0.00	0.62 ± 0.02	0 ± 0.00
Data7	0.0 ± 0.00	1.01 ± 0.02	1.12 ± 0.012
Data8	0.0 ± 0.00	1.1 ± 0.03	1.21 ± 0.01
Pat	0.85 ± 0.00	1.25 ± 0.002	0.85 ± 0.00
Iris	0.62 ± 0.02	0.847726 ± 0.01	0.854081 ± 0.025
Cancer	0.367056 ± 0.001	0.386768 ± 0.02	0.380332 ± 0.023
Newthyroid	0.58 ± 0.03	0.828616 ± 0.021	0.838885 ± 0.022
Glass	1.106217 ± 0.01	1.324295 ± 0.022	1.117940 ± 0.023
LiverDisorder	0.987329 ± 0.01	0.982611 ± 0.03	0.981873 ± 0.021

is able to determine the proper cluster number only for *Data1*, *Data4*, *Data5*, *Pat*, *Cancer* and *LiverDisorder* data sets. However, for *Data1* and *Pat* even though GCUK-clustering is able to detect the proper number of clusters, the final partitionings identified by it (shown in Figure 7(b) and 14(b)) are not proper. Figures 7(c), 8(c), 9(c), 10(c), 11(a) 11(b), 12(c), 13(c) and 14(c) show, respectively, the clustering results obtained after application of HNGA-clustering on the nine artificial data sets. Again, results shown in Table I, reveal that HNGA-clustering is able to determine the proper cluster number only for *Data4*, *Data5*, *Data6*, *Cancer*, *Glass* and *LiverDisorder* data sets. Thus it is easy to conclude that HNGA-clustering is only able to find out hyperspherical clusters from a data set but not any other shaped clusters. The main reason behind such performance is that it optimizes a convex combination of some cluster validity indices all of which are only able to detect hyperspherical shaped clusters. For three real-life data sets, the Hinton diagrams representing similarities of all pairs of actual and obtained cluster centers by GCUK-clustering and HNGA-clustering algorithms according to Euclidean distance are also shown in Figures 18-20. The 2-d projection of the segmentation results obtained by GCUK-clustering and HNGA-clustering for three real-life data sets are also shown in Figures 15-17. *Minkowski Score* (MS) [39] of the resultant partitioning is calculated after application of all the three algorithms on all the data sets used here for experiment. MS is a measure of the quality of a solution given the true clustering . Let T be the “true” solution and S the solution we wish to measure. Denote by n_{11} the number of pairs of elements that are in the same cluster in both S and T. Denote by n_{01} the number of pairs that are in the same cluster only in S, and by n_{10} the number of pairs that are in the same cluster in T. *Minkowski Score* (MS) is then defined as:

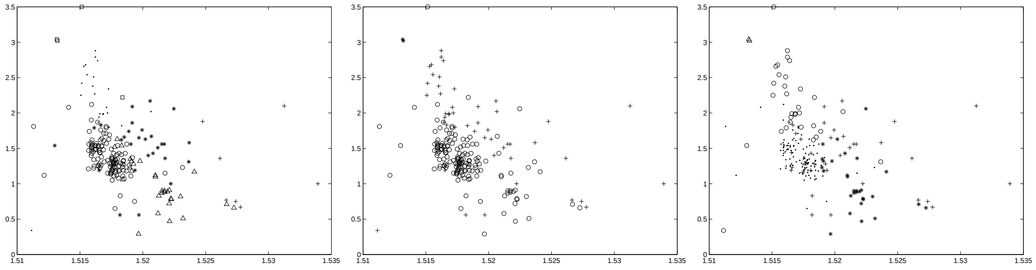
$$MS(T,S) = \sqrt{\frac{n_{01}+n_{10}}{n_{11}+n_{10}}}. \tag{19}$$

For MS, the optimum score is 0, with lower scores being “better”. Each of the above mentioned three algorithms are executed ten times for each of the data sets. The average MS scores and their standard deviations for all the experimental data sets after application of the three algorithms are given in Table II. For all the data sets, except for *Data4* and *LiverDisorder*, VGAPS-clustering is found to provide low MS which indicates that partitioning corresponding to VGAPS-clustering is the best among the three clustering algorithms. ANOVA [40] statistical analysis is performed on the combined results of the three algorithms. The One-Way ANOVA procedure produces a one-way analysis of variance for a quantitative dependent variable (here it is MS value) by a single independent variable (here it is the algorithm). Analysis of variance is used to test the hypothesis that several means are equal. From the statistical test ANOVA, it is found



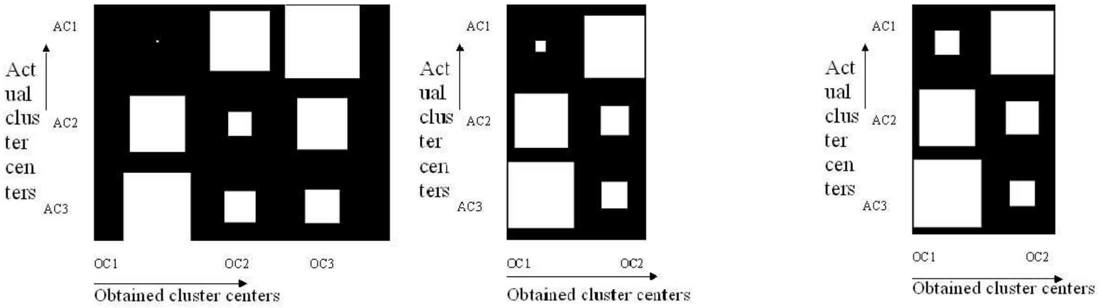
(a) (b) (c)

Fig. 16. Clustered *Newthyroid* data projected on the first two dimensions using (a) VGAPS-clustering where 3 clusters are detected (b) GCUK-clustering where 8 clusters are detected (c) HNGA-clustering where 5 clusters are detected



(a) (b) (c)

Fig. 17. Clustered *Glass* data projected on the first and the fourth dimensions using (a) VGAPS-clustering where 6 clusters are detected (b) GCUK-clustering where 3 clusters are detected (c) HNGA-clustering where 6 clusters are detected



(a) (b) (c)

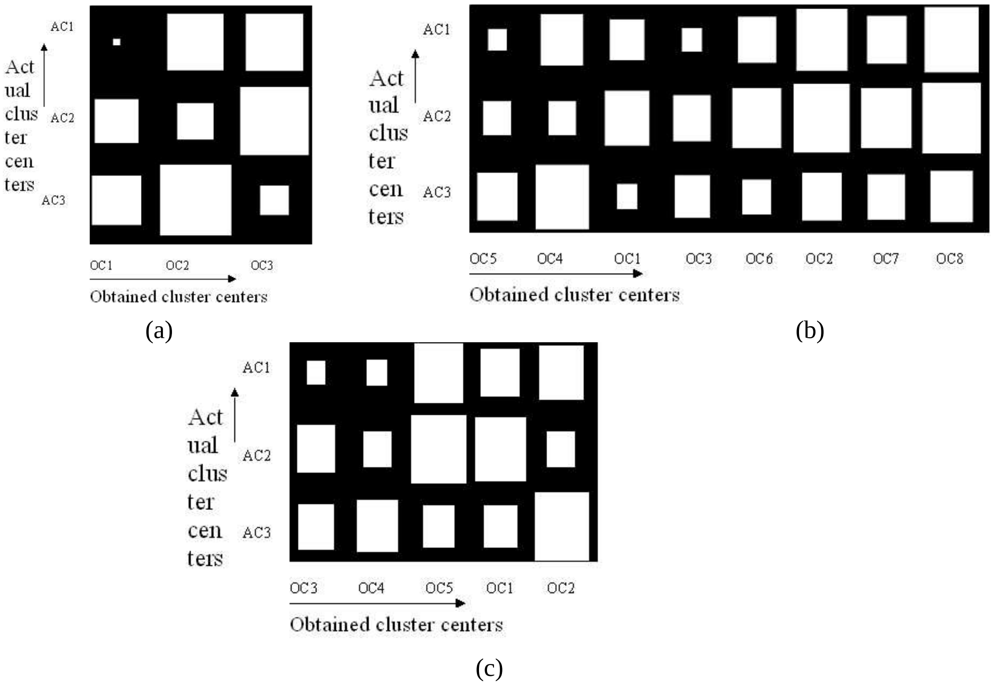
Fig. 18. Hinton diagram showing the similarities in terms of Euclidean distance between all pairs of actual and observed cluster centers for *Iris* data set by (a) VGAPS-clustering (b) GCUK-clustering (c) HNGA-clustering techniques. Here OC denotes obtained cluster centers and AC denotes actual cluster centers.

that the difference in the mean MS values obtained by VGAPS-clustering with those obtained by GCUK-clustering and HNGA-clustering algorithms, are statistically significant with significance value < 0.05 for all data sets except *Data4* and *LiverDisorder*. This indicates that the better performance of VGAPS as compared to GCUK-clustering and HNGA-clustering for all the data sets except *Data4* and *LiverDisorder*, in terms of the MS scores, are statistically significant. For *Data4*, HNGA-clustering performs the best in terms of MS score and the difference in the mean MS values is statistically significant with significance value $2.4603e-008$. For *LiverDisorder* although the mean MS score obtained by VGAPS-clustering is poorer than those obtained by HNGA-clustering and GCUK-clustering, the difference in the mean MS values are not statistically significant (here significance values are 0.42 and 0.34, respectively).

It may be noted that while assigning points using point symmetry based distance in VGAPS-clustering, every reflected point may not be close to another true point. We have also counted how many points are really assigned to different clusters based on the point symmetry based distance rather than the Euclidean distance while VGAPS is executed on different data sets. Our experiments show that for *Data1*, *Data3*, *Data4* and *Iris* data sets, 90%, 75%, 86% and 98% points are assigned to different clusters based on the point symmetry based distance.

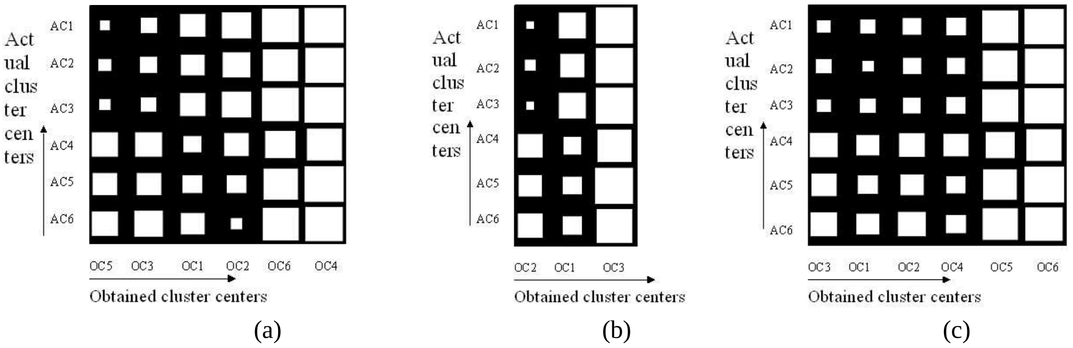
C. *Effectiveness of Using Kd-tree for Nearest Neighbor Search*

Note that the proposed computation of the *Sym*-index utilizes a Kd-tree structure to reduce the time required for identifying the nearest neighbors. In order to demonstrate the computational advantage thus obtained, VGAPS-clustering is executed without using the Kd-tree data structure on a PIV processor, 1.6GHz speed, running Linux. Table III provides the time required for the two cases for three data sets, namely, *Data1*, *Data4* and *Data5*.



(a) (b) (c)

Fig. 19. Hinton diagram showing the similarities in terms of Euclidean distance between all pairs of actual and observed cluster centers for *Newthyroid* data set by (a) VGAPS-clustering (b) GCUK-clustering (c) HNGA-clustering techniques. Here OC denotes obtained cluster centers and AC denotes actual cluster centers.



(a) (b) (c)

Fig. 20. Hinton diagram showing the similarities in terms of Euclidean distance between all pairs of actual and observed cluster centers for *Glass* data set by (a) VGAPS-clustering (b) GCUK-clustering (c) HNGA-clustering techniques. Here OC denotes obtained cluster centers and AC denotes actual cluster centers.

As is evident, incorporation of Kd-tree significantly reduces the computational burden of the process. The time taken by the GCUK and HNGA clustering algorithms for these datasets are also reported in Table III. It is evident from this table that the computation of point symmetry based distance makes the proposed VGAPS-clustering more time consuming than the existing algorithms for unknown K. It is also well-known that the Kd-tree is not efficient for high dimensional data ($d > 20$). In order to investigate the time required for some high dimensional data sets, a 20-dimensional artificial data set is generated consisting of 3000 points. The data has three clusters, two hyperellipsoidal shaped clusters and one hyperspherical cluster (containing 1000 points in each). VGAPS-clustering with Kd-tree took 6 minutes to execute the first generation where as VGAPS-clustering without Kd-tree didn't complete its first generation even in 250 minutes. Similar data sets are also generated by increasing the number of dimensions to 30, 40, 50, 60 and 70, respectively. The total time taken by VGAPS-clustering with Kd-tree to execute one generation for these data sets are 16, 17, 36, 47 and 54 minutes, respectively. Thus, the time taken by VGAPS-clustering for higher dimensional data sets is quite significant. However, its ability to detect the number of clusters and the proper partitioning from data sets having clusters of widely varying characteristics, irrespective

TABLE III

EXECUTION TIME (VGAPS IS IMPLEMENTED IN C AND EXECUTED ON LINUX PLATFORM, PIV PROCESSOR, 1.66 GHZ SPEED) IN SECONDS BY VGAPS WITH AND WITHOUT KD TREE BASED SEARCH, GCUK-CLUSTERING AND HNGA-CLUSTERING

Data set	VGAPS with Kd tree	VGAPS with out Kd tree	GCUK	HNGA
Data1	77	5280	68	72
Data4	62	2268	60	61
Data5	128	6112	62.353	85

of their convexity, or overlap or size, as long as they possess the property of symmetry, might offset this limitation for situations where clustering performance not time is the most overriding consideration.

VI. Discussion and Conclusion

Most of the clustering methods make prior assumptions about the structure of the clusters. For example, GCUK-clustering, which is a genetic K-means technique for automatic determination of clusters, can only detect equisized hyperspherical clusters from a data set. In this article a new point symmetry based distance is utilized to develop a variable string length genetic clustering technique (VGAPS-clustering) which automatically evolves the number of clusters present in a data set. The newly proposed cluster validity index, *Sym*-index, which is capable of detecting both the proper partitioning and the proper number of clusters present in a data set, is used as the fitness of the chromosomes. In VGAPS-clustering, the assignment of points to different clusters is done based on the point symmetry distance rather than the Euclidean distance when the point is indeed symmetric with respect to a center. Moreover, the use of adaptive mutation and crossover probabilities helps VGAPS-clustering to converge faster. Kd-tree based nearest neighbor search is utilized to reduce the computational complexity of computing the point symmetry based distance. The global convergence property of the proposed VGAPS-clustering is also established. The effectiveness of the VGAPS-clustering, as compared to two recently proposed automatic clustering techniques, namely, GCUK-clustering and HNGA-clustering, is demonstrated on nine artificially generated and five real-life data sets of different characteristics. Results on the fourteen data sets establish the fact that VGAPS-clustering is well-suited to detect the number of clusters and the proper partitioning from data sets having clusters of widely varying characteristics, irrespective of their convexity, or overlap or size, as long as they possess the property of symmetry. VGAPS seeks for clusters which are point symmetric with respect to their centers. Thus VGAPS will fail if the clusters do not have this property. Based on these observations, and the fact that the property of symmetry is widely evident in real-life situations, application of VGAPS-clustering to automatically determine the proper number of clusters and the proper partitioning from different data sets seems justified and is therefore recommended.

The current work concentrates only on a particular form of symmetry viz., point-based symmetry. Other forms of symmetry may be line-based symmetry, polynomial symmetry etc. Techniques for detecting clusters, along with their theoretical analysis, with these forms of symmetry need to be developed in the future. The application of VGAPS-clustering for medical image segmentation as well as object detection in images is another direction of future work. Finally, development of some multiobjective clustering technique using symmetry, connectivity, compactness etc. as different objective functions so that it can work well for partitioning any type of data sets needs to be investigated.

References

[1] B. S. Everitt, S. Landau, and M. Leese, *Cluster Analysis*. London: Arnold, 2001.

[2] F. Attneave, "Symmetry information and memory for pattern," *Am. J. Psychology*, vol. 68, pp. 209–222, 1995.

[3] S. Bandyopadhyay and S. Saha, "GAPS: A clustering method using a new point symmetry based distance measure," *Pattern Recog.*, vol. 40, pp. 3430–3451, 2007.

[4] M.-C. Su and C.-H. Chou, "A modified version of the k-means algorithm with a distance based on cluster symmetry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 6, pp. 674–680, 2001.

[5] M. R. Anderberg, *Computational Geometry: Algorithms and Applications*. Springer, 2000.

[6] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum, 1981.

[7] I. Gath and A. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 773–781, 1989.

[8] R. Krishnapuram and C. P. Freg, "Fitting an unknown number of lines and planes to image data through compatible cluster merging," *Pattern Recognition*, vol. 25, no. 4, pp. 433–439, 1992.

[9] S. Bandyopadhyay and U. Maulik, "Nonparametric genetic clustering: Comparison of validity indices," *IEEE Transactions On Systems, Man and Cybernetics, Part C*, vol. 31, no. 1, pp. 120–125, 2001.

[10] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 12, pp. 1650–1654, 2002.

[11] H. C. Chou, M. C. Su, and E. Lai, "A new cluster validity measure and its application to image compression," *Pattern Analysis and Applications*, vol. 7, pp. 205–220, July 2004.

[12] W. Wang and Y. Zhang, "On fuzzy cluster validity indices," *Fuzzy Sets and Systems*, vol. 158, no. 19, pp. 2095–2117, October, 2007.

[13] P. B. Helena Br$\acute{a}$s Silva and J. P. da Costa, "A partitional clustering algorithm validated by a clustering tendency index based on graph theory," *Pattern Recognition*, vol. 39, no. 5, pp. 776–788, May 2006.

[14] M. Kim and R. Ramakrishna, "New indices for cluster validity assessment," *Pattern Recognition Letters*, vol. 26, no. 15, pp. 2353–2363, November 2005.

[15] R. H. Eduardo and F. F. E. Nelson, "A genetic algorithm for cluster analysis," *Intelligent Data Analysis*, vol. 7, pp. 15–25, 2003.

[16] S. Bandyopadhyay and U. Maulik, "Genetic clustering for automatic evolution of clusters and application to image classification," *Pattern Recognition*, no. 2, pp. 1197–1208, 2002.

[17] J. H. Holland, *Adaptation in Natural and Artificial Systems*. AnnArbor: The University of Michigan Press, 1975.

[18] W. Sheng, S. Swift, L. Zhang, and X. Liu, "A weighted sum validity function for clustering with a hybrid niching genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 35, no. 6, December, 2005.

[19] S. Bandyopadhyay, U. Maulik, and A. Mukhopadhyay, "Multiobjective genetic clustering for pixel classification in remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1506–1511, 2007.

[20] A. K. Jain, P. Duin, and M. Jianchang, "Statistical pattern recognition : A review," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[21] S. Bandyopadhyay, "Simulated annealing using reversible jump markov chain monte carlo algorithm for fuzzy clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 479–490, 2005.

[22] S. Saha and S. Bandyopadhyay, "A fuzzy genetic clustering technique using a new symmetry based distance for automatic evolution of clusters," in *ICCTA*, pp. 309–314, 2007.

[23] D. M. Mount and S. Arya, "ANN: A library for approximate nearest neighbor searching," 2005. http://www.cs.umd.edu/∼mount/ANN.

[24] U. Maulik and S. Bandyopadhyay, "Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 5, pp. 1075– 1081, 2003.

[25] M. Srinivas and L. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man and Cybernatics*, vol. 24, no. 4, pp. 656–667, April, 1994.

[26] G. Rudolph, "Convergence analysis of canonical genetic algorithms," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 96–101, 1994.

[27] K. Krishna and M. N. Murty, "Genetic k-means algorithm," *IEEE Transactions on Systems, Man, And Cybernetics-Part B*, vol. 29, no. 3, pp. 433–439, June,1999.

[28] A. M. Bensaid, L. O. Hall, J. C. Bezdek, L. P. Clarke, M. L. Silbiger, J. A. Arrington, and R. F. Murtagh, "Validity-guided (re)clustering with applications to image segmentation," *IEEE Transactions on Fuzzy Systems*, vol. 4, no. 2, pp. 112–123, 1996.

[29] S. Mitra and S. K. Pal, "Fuzzy multi-layer perceptron, inferencing and rule generation," *IEEE Transactions on Neural Networks*, vol. 6, pp. 51–63, 1995.

[30] D. N. A. Asuncion, "UCI machine learning repository," 2007.

[31] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 3, pp. 179–188, 1936.

[32] M. D. Berg, M. V. Kreveld, M. Overmars, and O. Schwarzkopf, *Cluster Analysis for Application*. Academic Press, 1973.

[33] C. H. Chou, M. C. Su, and E. Lai, "Symmetry as a new measure for cluster validity," in *2nd WSEAS Int. Conf. on Scientific Computation and Soft Computing*, pp. 209–213, 2002.

[34] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 1, pp. 224–227, 1979.

[35] L. Kaufman and P. Rousseuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. NY, US: John Wiley & Sons, 1990.

[36] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *J. Cyberns.*, vol. 3, pp. 32–57, 1973.

[37] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," *IEEE Transactions on Systems, Man, And Cybernetics*, vol. 28, pp. 301–315, 1998.

[38] R. B. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Comm. in Stat.*, vol. 3, pp. 1–27, 1974.

[39] A. Ben-Hur and I. Guyon, *Detecting Stable Clusters using Principal Component Analysis in Methods in Molecular Biology*. Humana press, 2003.

[40] T. W. Anderson and S. Scolve, *Introduction to the Statistical Analysis of Data*. Houghton Mifflin, 1978.

Sanghamitra Bandyopadhyay (SM'05) did her BS, MS and Ph. D. in Computer Science in 1991, 1993 and 1998 respectively. Currently she is an Associate Professor at Indian Statistical Institute, India. She has worked in Los Alamos National Laboratory, Los Alamos, USA, University of New South Wales, Sydney, Australia, University of Texas at Arlington, USA, University of Maryland at Baltimore, USA, Fraunhofer Institute, Germany, and Tsinghua University, China. Dr. Bandyopadhyay is the first recipient of Dr. Shanker Dayal Sharma Gold Medal and also the Institute Silver Medal for being adjudged the best all round post-graduate performer in IIT, Kharagpur, India, in 1994. She has also received the Young Scientist Awards of the Indian National Science Academy (INSA) and the Indian Science Congress Association (ISCA) in 2000. In 2002 Dr. Bandyopadhyay received the Young Scientist Awards of the Indian National Academy of Engineers (INAE). Dr. Bandyopadhyay was an invited speaker at the 8th International Conference on Human and Computers 2005, held in Aizu, Japan during 30th August to 2nd September 2005. She is a senior member of Institute of Electrical and Electronics Engineers (IEEE). Dr. Bandyopadhyay has co-authored more than one hundred and twenty five technical articles in international journals, book chapters and conference/workshop proceedings. She has delivered many invited talks and tutorials around the world. She was the Program Co-Chair of the First International Conference on Pattern Recognition and Machine Intelligence, (PReMI'05) held in Kolkata, India, during December 18-22, 2005. She has recently published an authored book titled "*Classification and Learning Using Genetic Algorithms: Applications in Bioinformatics and Web Intelligence*" from Springer and two edited books titled "*Advanced Methods for Knowledge Discovery from Complex Data*", published by Springer, UK in 2005, and "*Analysis of Biological Data: A Soft Computing Approach*" published by World Scientific in 2007. She has also edited journals special issues in the area of Soft Computing, Data Mining and Bioinformatics. Her research interests include Computational Biology and Bioinformatics, Soft and Evolutionary Computation, and Image Processing. Pattern Recognition and Data Mining.

Sriparna Saha received her B.Tech degree in Computer Science and Engineering from Kalyani Govt. Engineering College, University of Kalyani, India in 2003. She did her M.Tech in Computer Science from Indian Statistical Institute, Kolkata in 2005. She is the recipient of Lt Rashi Roy Memorial Gold Medal from Indian Statistical Institute for outstanding performance in M.Tech (Computer Science). At present she is pursuing her Ph.D. from Indian Statistical Institute, Kolkata, India. She has co-authored more than twenty articles in international journals and conference/workshop proceedings. She is the recipient of "Google India Women In Engineering Award, 2008". Her research interests include Multiobjective Optimization, Pattern Recognition, Evolutionary Algorithms, and Data Mining.