Information Sciences 146 (2002) 221–237

An evolutionary technique based on K-Means algorithm for optimal clustering in $\mathbb{R}^N$

Sanghamitra Bandyopadhyay [a,*], Ujjwal Maulik [b]

[a] *Machine Intelligence Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700108, India*
[b] *Department of Computer Science and Engineering, Kalyani Government Engineering College, Kalyani, Nadia, India*

Received 8 March 2000; received in revised form 30 December 2001; accepted 19 April 2002

Abstract

A genetic algorithm-based efficient clustering technique that utilizes the principles of K-Means algorithm is described in this paper. The algorithm called KGA-clustering, while exploiting the searching capability of K-Means, avoids its major limitation of getting stuck at locally optimal values. Its superiority over the K-Means algorithm and another genetic algorithm-based clustering method, is extensively demonstrated for several artificial and real life data sets. A real life application of the KGA-clustering in classifying the pixels of a satellite image of a part of the city of Mumbai is provided.

Keywords: Clustering; Genetic algorithms; K-Means algorithm; Satellite image classification

1. Introduction

Clustering [1–7] is an important unsupervised classification technique used in identifying some inherent structure present in a set of objects. The purpose of cluster analysis is to classify objects into subsets that have some meaning in

* Corresponding author. Fax: +91-33-577-6680.
E-mail addresses: sanghami@isical.ac.in (S. Bandyopadhyay), ujjwal_maulik@kucse.wb.nic.in (U. Maulik).

the context of a particular problem. More specifically, in clustering, a set of patterns, usually vectors in a multi-dimensional space, are grouped into clusters in such a way that patterns in the same cluster are similar in some sense and patterns in different clusters are dissimilar in the same sense.

In some clustering problems, the number of clusters, K, is known a priori. In such situations clustering may be formulated as distribution of n patterns in N dimensional metric space among K groups, such that the patterns in a group are more similar to each other than to patterns in different groups. This involves minimization of some extrinsic optimization criterion. K-Means algorithm is a well known and widely used clustering technique applicable in such situations. However, the major drawback of the K-Means algorithm is that it often gets stuck at local minima and the result is largely dependent on the choice of the initial cluster centers. An attempt is made in this paper to integrate the effectiveness of the K-Means algorithm for partitioning data into a number of clusters, with the capability of genetic algorithm for providing the requisite perturbation to bring it out of this local minima.

Genetic algorithms (GAs) [8–10] are randomized search and optimization techniques guided by the principles of evolution and natural genetics. They are efficient, adaptive and robust search processes, performing multi-dimensional search in order to provide near optimal solutions of an evaluation (fitness) function in an optimization problem. Since the problem of clustering may be viewed as searching for a number of clusters in the feature space such that a given clustering metric is optimized, application of GAs to this problem seems natural and appropriate. One such attempt can be found in [11].

Murthy and Chowdhury [11] have considered a partition to be encoded as a string of length n, where n is the number of data points. The ith element of the chromosome represents the cluster number to which the corresponding point belongs. A comparison of the performance of their algorithm, subsequently referred to as the GA-clustering algorithm, with that of the K-Means algorithm is provided in [11]. Note that with the increase in the string length, the search space in GAs increases; thereby making the process more time consuming. Hence, when the number of points to be clustered is very large, which may happen in many real life situations, the method proposed in [11] (where the size of a chromosome is equal to the number of data points) will have limited applicability.

In this paper, we describe a GA-based clustering algorithm where the chromosome encodes the centers of the clusters instead of a possible partition of the data points. (Note that the length of a chromosome in the algorithm is restricted by the number of clusters, rather than the number of data points as in [11].) The algorithm tries to evolve the appropriate cluster centers while optimizing a given clustering metric.

Traditionally, GAs assume no prior knowledge of the problem under consideration. The only step that requires such a knowledge is the fitness com-

putation procedure. They are therefore applicable to a wide variety of problems. However, in many situations, some additional information about the search space is often available. These can be effectively incorporated in GAs for improving its searching capability [12]. In the domain of clustering, it is often assumed that the centroid of the points belonging to a cluster represents the center of that cluster. This knowledge is incorporated in the fitness evaluation process of the GA-based clustering method, thereby providing the KGA-clustering algorithm, to make the search more efficient. The details are presented in Section 3.1.

Experimental results comparing the performance of the proposed KGA-clustering method with those of the K-Means and the GA-clustering [11] algorithms are provided for several artificial and real-life data sets. Moreover, the utility of the KGA-clustering algorithm for pixel classification of a satellite image for differentiating different land-cover regions is demonstrated. Note that although GAs usually deal with binary strings, we have implemented floating point coding of the chromosomes since it is a more natural form of representation for this problem.

2. Clustering

In this section, we first provide a formal statement of the clustering problem. Since we have compared the performance of our KGA-clustering algorithm with that of the K-Means algorithm, a brief outline of the latter is also provided.

2.1. Basic principle

Clustering in N dimensional Euclidean space $\mathbb{R}^N$ is the process of partitioning a given set of n points into a number, say K, of groups (or, clusters) based on some similarity/dissimilarity metric which establishes a rule for assigning patterns to the domain of a particular cluster center. Let the set of n points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ be represented by the set S and the K clusters be represented by $C_1, C_2, \ldots, C_K$. Then

$$C_i \neq \emptyset \quad \text{for } i = 1, \ldots, K,$$

$$C_i \cap C_j = \emptyset \quad \text{for } i = 1, \ldots, K,\ j = 1, \ldots, K \text{ and } i \neq j,$$

$$\bigcup_{i=1}^{K} C_i = S.$$

2.2. Clustering by K-Means algorithm

K-Means [3] is one of the widely used clustering techniques, which is an iterative hill climbing algorithm. It consists of the following steps:

Step 1. Choose K initial cluster centers $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_K$ randomly from the n points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.

Step 2. Assign point $\mathbf{x}_i$, $i = 1, 2, \ldots, n$ to cluster C_j, $j \in \{1, 2, \ldots, K\}$ iff

$$\|\mathbf{x}_i - \mathbf{z}_j\| \leqslant \|\mathbf{x}_i - \mathbf{z}_p\|, \quad p = 1, 2, \ldots, K, \text{ and } j \neq p.$$

Ties are resolved arbitrarily.

Step 3. Compute new cluster centers $\mathbf{z}_1^*, \mathbf{z}_2^*, \ldots, \mathbf{z}_K^*$, as follows:

$$\mathbf{z}_i^* = \frac{1}{n_i} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j, \quad i = 1, 2, \ldots, K,$$

where n_i is the number of elements belonging to cluster C_i.

Step 4. If $\mathbf{z}_i^* = \mathbf{z}_i \ \forall i = 1, 2, \ldots, K$ then terminate. Otherwise continue from step 2.

Note that the process is executed for a predetermined fixed number of iterations unless it terminates at step 4.

Although K-Means is one of the widely used clustering techniques, it is known that the solution it provides depends on the choice of the initial cluster centers [3]. Also it has been shown in [13] that the algorithm may fail to converge to a local minimum under certain conditions. Moreover, global solutions of large problems cannot be found with a reasonable amount of computation effort [14]. It is because of these factors that several approximate methods are developed to solve the underlying optimization problem. As described in the next section, GA is one such technique that may be efficiently applied for finding optimal clusters by minimizing the extrinsic clustering metric.

3. Clustering using GAs

The algorithm described in this section has been designed for use in the areas where K-Means algorithm has wide spread applicability. The KGA clustering algorithm is first described in detail followed by a brief outline of the algorithm of Murthy and Chowdhury [11].

3.1. KGA-clustering algorithm

The KGA-clustering algorithm appropriately determines K cluster centers in $\mathbb{R}^N$; thereby suitably clustering the set of n unlabeled points. For N dimensional feature space, each chromosome is represented by a sequence of $N * K$ floating point numbers where the first N positions (or, genes) represent the N dimensions of the first cluster center, the next N positions represent those of the second cluster center, and so on. The sum of the Euclidean distances of the points from their respective cluster centers is taken as the clustering metric $\mathscr{M}$.

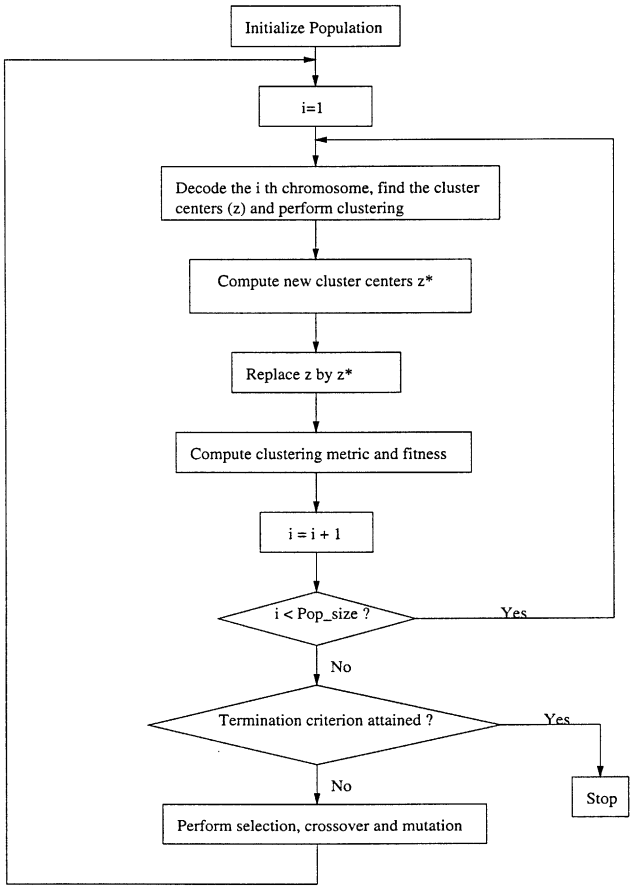

Fig. 1. Flowchart of the KGA-clustering algorithm.

The goal is to search for the appropriate cluster centers such that $\mathscr{M}$ is minimized. The flowchart of KGA-clustering algorithm is provided in Fig. 1. In each generation the principles of K-Means algorithm are utilized for determining new cluster centers. Subsequently, the old cluster centers are replaced by these new values thereby incorporating some problem specific knowledge in the system. However, this greedy characteristic of K-Means algorithm makes it liable to get stuck at locally optimal values. In order to avoid this, features of GAs are used for perturbing the system. The different steps of KGA-clustering algorithm are now described briefly.

3.1.1. Population initialization

K randomly chosen distinct points from the data set are used to initialize the K cluster centers encoded in each chromosome. This is similar to the

initialization of the cluster centers in K-Means algorithm. This process is repeated for each chromosome in the population.

3.1.2. Clustering

In this step, the clusters are formed according to the centers encoded in the chromosome. This is done by assigning each point $\mathbf{x}_i$, $i = 1, 2, \ldots, n$ to one of the clusters C_j with center $\mathbf{z}_i^*$ such that

$$\|\mathbf{x}_i - \mathbf{z}_j\| \leqslant \|\mathbf{x}_i - \mathbf{z}_p\|, \quad p = 1, 2, \ldots, K, \text{ and } p \neq j.$$

All ties are resolved arbitrarily. As like the K-Means algorithm, for each cluster C_i, its new center $\mathbf{z}_j^*$ is computed as

$$z_i^* = \frac{1}{n_i} \sum_{\mathbf{x}_j \in C_i} \mathbf{x}_j, \quad i = 1, 2, \ldots, K,$$

where n_i is the number of points in cluster C_i. These $\mathbf{z}_i^*$s now replace the previous $\mathbf{z}_i$s in the chromosome.

3.1.3. Fitness computation

For each chromosome, the clusters formed in the previous step are utilized for computing the clustering metric, $\mathscr{M}$, as follows:

$$\mathscr{M} = \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in C_i} \|\mathbf{x}_j - \mathbf{z}_i\|. \tag{1}$$

For finding the appropriate clusters $\mathscr{M}$ has to be minimized. The fitness function of a chromosome is defined as $1/\mathscr{M}$. Therefore, maximization of the fitness function will lead to minimization of the clustering metric $\mathscr{M}$.

3.1.4. Genetic operations

- *Selection*. The selection process selects chromosomes from the mating pool directed by the survival of the fittest concept of natural genetic systems. In the proportional selection strategy adopted in this paper, a chromosome is assigned a number of copies, which is proportional to its fitness in the population.
- *Crossover*. Crossover is a probabilistic process that exchanges information between two parent chromosomes for generating two offspring. Here, single-point crossover with a fixed crossover probability of μ_c is used. For chromosomes of length l ($l = NK$), a random integer, called the crossover point, is generated in the range $[1, l-1]$. The portions of the chromosomes lying to the right of the crossover point are exchanged to produce two offspring.
- *Mutation*. Each chromosome undergoes mutation with a fixed probability μ_m. Let $\mathscr{M}_{\min}$ and $\mathscr{M}_{\max}$ be the minimum and maximum values of the clus-

tering metric, respectively, in the current population. For mutating a chromosome whose clustering metric is $\mathscr{M}$, a number δ in the range $[-R, +R]$ is generated with uniform distribution, where

$$R = \begin{cases} \frac{\mathscr{M}-\mathscr{M}_{\min}}{\mathscr{M}_{\max}-\mathscr{M}_{\min}} & \text{if } \mathscr{M}_{\max} > \mathscr{M}, \\ 1 & \text{if } \mathscr{M}_{\min} = \mathscr{M}_{\max}. \end{cases}$$

If the minimum and maximum values of the data set along the ith dimension ($i = 1, 2, \ldots, N$) are $x^i_{\min}$ and $x^i_{\max}$, respectively, and the position to be mutated is the ith dimension of a cluster center with value x^i, then after mutation the value becomes

$$\begin{cases} x^i + \delta \times (x^i_{\max} - x^i) & \text{if } \delta \geqslant 0, \\ x^i + \delta \times (x^i - x^i_{\min}) & \text{otherwise.} \end{cases}$$

Note that this scheme of mutation provides perturbation in a maximum range to strings either when they have the largest value of $\mathscr{M}$ in the population (i.e., $\mathscr{M} = \mathscr{M}_{\max}$) or when all the strings have the same value of the clustering metric (i.e., $\mathscr{M} = \mathscr{M}_{\min} = \mathscr{M}_{\max}$). On the other hand, the best string(s) in the population (i.e., the one with $\mathscr{M} = \mathscr{M}_{\min}$) is not perturbed at all in the current generation. Moreover, the perturbation is such that the mutated centers still lie within the bounds of the data points.

3.1.5. Termination criterion

The processes of clustering, fitness computation, selection, crossover, and mutation are executed for a maximum number of iterations. The best string or chromosome seen up to the last generation provides the solution to the clustering problem. Elitism has been implemented in each generation by replacing the worst chromosome of the population with the best one seen up to the previous generation.

Since the performance of the KGA-clustering algorithm is compared to that of another GA-based clustering technique where the number of clusters is known a priori, a brief outline of the latter is now provided.

3.2. GA-clustering algorithm [11]

In GA-clustering [11], a possible partition is encoded in a chromosome of length n, where n is the number of data points. The ith element of the chromosome denotes the cluster number assigned to the point $\mathbf{x}_i$. For initializing the population, the chromosomes are generated randomly, where the value of each element in a chromosome is allowed to lie between 1 and K. Since the chromosome itself provides a partition of the data points, it is no longer necessary to determine the appropriate clusters as is done for KGA-clustering

algorithm. For computing the fitness, the centroid of the clusters are determined, and the clustering metric $\mathscr{M}$ is computed as in Eq. (1).

The mating pool is created using proportional selection scheme. Conventional single point crossover is applied on the chromosomes. Because of crossover, some invalid chromosomes, i.e., where all the clusters are not represented, may be generated. For example, if chromosomes (1 2 2 3 2 1) and (1 3 3 2 2 1) are crossed after the third position, then we get offspring (1 2 2 2 2 1) and (1 3 3 3 2 1). Here the first offspring is invalid since it does not contain any point from cluster 3. In case such invalid offspring are produced, the crossover process is repeated until valid offspring result, or a limit on the number of attempted crossovers is reached.

Mutation is performed occasionally, where a position in a chromosome is chosen at random, and its value is replaced by a number in the range $[1, K]$. Again, mutation may result in invalid strings which are tackled in a manner similar to that of the crossover operation. Elitist model of GAs is used for developing the clustering algorithm. The steps of fitness computation, selection, crossover and mutation are repeated for a fixed number of iterations, after which the best chromosome provides the final clustering.

The following section provides the results of implementation of the KGA-clustering algorithm, along with its comparison with the K-Means and GA-clustering [11] algorithms for several artificial and real life data sets.

4. Implementation results

Two artificial data sets (*Data 1* and *Data 2*) and three real-life data sets (*Vowel*, *Iris* and *Crude Oil*) are considered for the purpose of conducting the experiments. These data sets are first described below.

4.1. Data sets

4.1.1. Artificial data

Data 1. This is a two dimensional data set having 10 points where the number of clusters is two. Thus for this data set the value of K is chosen to be 2.

Data 2. This is a two dimensional data set having 76 points where the number of clusters is three. Thus for this data set the value of K is chosen to be 3.

4.1.2. Real life data sets

Vowel data. This data consists of 871 Indian Telugu vowel sounds [15]. These were uttered in a consonant–vowel–consonant context by three male speakers in the age group of 30–35 years. The data set has three features F_1, F_2, and F_3, corresponding to the first, second, and third vowel formant frequencies,

and six classes $\{\delta, a, i, u, e, o\}$. The value of K is therefore chosen to be 6 for this data.

Iris data. This data represents different categories of irises having four feature values. The four feature values represent the sepal length, sepal width, petal length and the petal width in centimeters [16]. It has three classes with 50 samples per class. The value of K is therefore chosen to be three for this data.

Crude oil data. This data [17] has 56 data points, five features and three classes. Hence the value of K is chosen to be 3 for this data set.

4.2. Comparative performance of the clustering algorithms

In this section, the performances of the KGA-clustering, GA-clustering [11] and K-Means algorithms are first compared in terms of the best value obtained after a fixed number of iterations. The two GA-based algorithms are also compared in terms of the number of iterations required to obtain a desired value. The crossover and mutation probabilities are $\mu_c = 0.8$ and $\mu_m = 0.001$, respectively. The population size is taken to be 10 for *Data 1* since it is a very simple data set, while it is taken to be 50 for the other data sets.

4.2.1. Clustering metric obtained after fixed number of iterations

For K-Means algorithm, as well as the two GA-based algorithms, we have fixed a maximum of 1000 iterations. However, it was observed that in all the experiments the K-Means algorithm terminated much before 1000 iterations.

The best, worst and the average values of the clustering metric ($\mathscr{M}_{\text{best}}$, $\mathscr{M}_{\text{worst}}$ and $\mathscr{M}_{\text{avg}}$, respectively) obtained for 50 distinct runs of the K-Means, KGA-clustering and GA-clustering [11] algorithms are shown in Table 1 for *Data 1*. Both the GA-based algorithms provide the optimum value of 2.225498. K-Means algorithm is found to sometimes get stuck at sub optimal values depending on the points that are chosen as the initial cluster centers.

Table 2 provides the corresponding values of $\mathscr{M}$ obtained by the three techniques for *Data 2*. The optimum value is 47.616026, which is obtained in all the runs of KGA-clustering algorithm. Noticeably, the algorithm of Murthy and Chowdhury fails to attain this value even once within 1000 generations. K-Means algorithm is found to attain this value in 22 of its 50 runs.

Table 1
$\mathscr{M}_{\text{best}}$, $\mathscr{M}_{\text{worst}}$ and $\mathscr{M}_{\text{avg}}$ obtained by K-Means, KGA-clustering and GA-clustering [11] algorithms for 50 different runs for *Data 1* when $K = 2$

	K-Means	KGA-clustering	GA-clustering
$\mathscr{M}_{\text{best}}$	2.225498	2.225498	2.225498
$\mathscr{M}_{\text{rest}}$	5.383182	2.225498	2.225498
$\mathscr{M}_{\text{avg}}$	3.488572	2.225498	2.225498

Table 2
$\mathcal{M}_{best}$, $\mathcal{M}_{worst}$ and $\mathcal{M}_{avg}$ obtained by K-Means, KGA-clustering and GA-clustering [11] algorithms for 50 different runs for *Data 2* when $K = 3$

	K-Means	KGA-clustering	GA-clustering
$\mathcal{M}_{best}$	47.616026	47.616026	60.032942
$\mathcal{M}_{worst}$	61.613077	47.616026	72.342458
$\mathcal{M}_{avg}$	57.713124	47.616026	66.800033

Table 3
$\mathcal{M}_{best}$, $\mathcal{M}_{worst}$ and $\mathcal{M}_{avg}$ obtained by K-Means, KGA-clustering and GA-clustering [11] algorithms for 50 different runs for *Vowel* when $K = 6$

	K-Means	KGA-clustering	GA-clustering
$\mathcal{M}_{best}$	149373.097180	149356.012425	383484.152448
$\mathcal{M}_{worst}$	151605.600107	149378.029757	395267.339008
$\mathcal{M}_{avg}$	149903.869317	149368.454400	390088.243508

Table 4
$\mathcal{M}_{best}$, $\mathcal{M}_{worst}$ and $\mathcal{M}_{avg}$ obtained by K-Means, KGA-clustering and GA-clustering [11] algorithms for 50 different runs for *Iris* when $K = 3$

	K-Means	KGA-clustering	GA-clustering
$\mathcal{M}_{best}$	97.204574	97.100777	124.127458
$\mathcal{M}_{worst}$	124.022373	97.100777	139.778272
$\mathcal{M}_{avg}$	107.724549	97.100777	135.404799

Tables 3 and 4 provide the best, worst and average values of $\mathcal{M}$ obtained by the three techniques for *Vowel* and *Iris*, respectively. As seen from the results, the results of the KGA-clustering algorithm are far superior to that of both K-Means and the GA-clustering algorithm of Murthy and Chowdhury. In fact, the GA-clustering algorithm is unable to provide meaningful clusters within the prescribed 1000 iterations for *Vowel*. Even for *Iris*, its result is found to be significantly poor.

Finally, Table 5 provides the best, worst and average values of $\mathcal{M}$ obtained by the three techniques for *Crude Oil*. It is found that the KGA-clustering algorithm is able to provide the same partition of the data points in all the runs. As earlier, the results of the other two methods are inferior to that of ours.

Table 5
$\mathcal{M}_{best}$, $\mathcal{M}_{worst}$ and $\mathcal{M}_{avg}$ obtained by K-Means, KGA-clustering and GA-clustering [11] algorithms for 50 different runs for *Crude Oil* when $K = 3$

	K-Means	KGA-clustering	GA-clustering
$\mathcal{M}_{best}$	279.484810	278.965150	297.048873
$\mathcal{M}_{worst}$	279.743216	278.965150	318.966627
$\mathcal{M}_{avg}$	279.597091	278.965150	308.155902

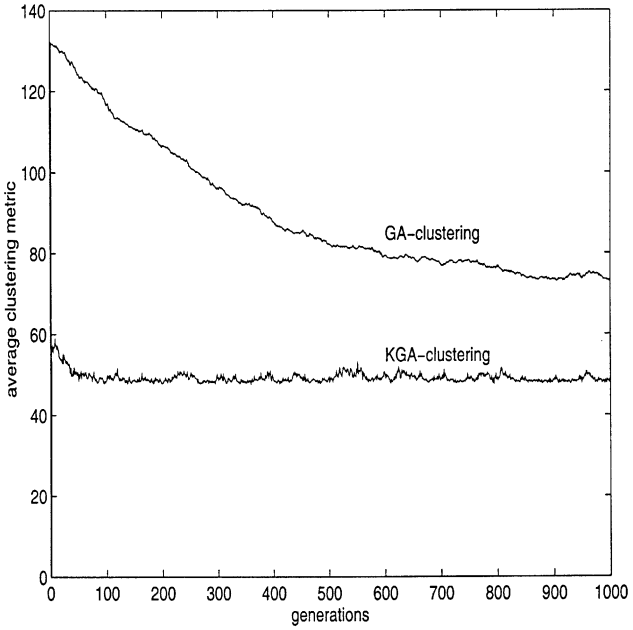

Fig. 2. Variation of the average clustering metric with the number of generations for *Data 2*.

In all the cases, the performance of the KGA-clustering algorithm is seen to be the best among the three methods. The performance of the GA-clustering algorithm of Murthy and Chowdhury is found to be poorer than that of even the K-Means algorithm (except for *Data 1*). Note that, the performance of the GA-based clustering methods may have further improved if more iterations are executed. For example, in the case of *Crude Oil* all the three values $\mathscr{M}_{\text{best}}$, $\mathscr{M}_{\text{worst}}$ and $\mathscr{M}_{\text{avg}}$ for the algorithm of Murthy and Chowdhury after 10,000 iterations became 278.965150. This conforms to the finding in [11], where this value was provided within 10,000 iterations in all the runs.

For comparing the performance of the two GA-based clustering techniques graphically, we have considered Figs. 2 and 3. The figures present the variation of the average clustering metric for *Data 2* and *Iris*, respectively in the first 1000 generations. Note that these figures, which reflect the overall performance of the algorithms, again demonstrate the significant superiority of the KGA-clustering over GA-clustering technique.

4.3. Number of iterations required to obtain a desired value

In this section, we compare the speed of convergence of the two GA-based clustering techniques to some desired value of the clustering metric, set in accordance with the best values obtained in the previous experiment for *Data 1*,

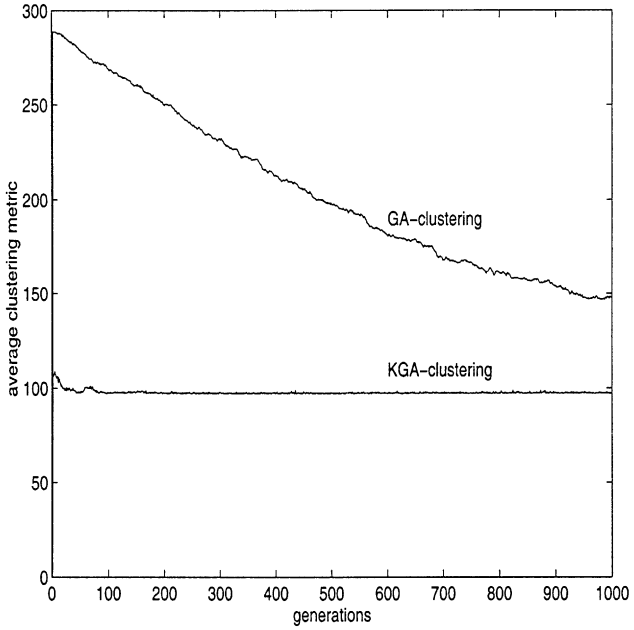

Fig. 3. Variation of the average clustering metric with the number of generations for *Iris*.

Table 6
Number of iterations required to obtain a desired value for KGA-clustering and GA-clustering algorithm [11]

Data set	Desired value	GA-clustering		KGA-clustering	
		Times obtained	Avg. number of iterations	Times obtained	Avg. number of iterations
Data 1	2.225498	50	21.0	50	2.4
Data 2	51.013294	0	–	50	3.0
Vowel	149356.012425	0	–	50	2115.0
Iris	97.100777	32	5868.0	50	357.6
Crude Oil	278.965150	50	4703.2	50	6.6

Data 2, *Vowel*, *Iris*, and *Crude Oil* data sets. The results are summarized in Table 6, where the entries are averaged over 50 runs of the algorithms when they are allowed to execute for a maximum of 10,000 iterations. As seen from the table, the GA-clustering algorithm is able to arrive at the desired value for *Data 1* and *Crude Oil* all the 50 times, while for *Iris* it does so 32 times. For the other two data sets, it is not able to provide the desired value even once. The corresponding entries in the column for 'Avg. number of iterations' are

therefore put as '–'. On the other hand, the KGA-clustering algorithm is able to obtain the desired value every time, in relatively fewer number of iterations. From the results it appears that the algorithm of Murthy and Chowdhury is significantly dependent on the number of data points, n. Their method is able to provide good partitions for data sets where n is comparatively small, (e.g., 10 for *Data 1*, 56 for *Crude Oil*) than for the cases where n is large (e.g., 871 for *Vowel*). This is expected since the length of a chromosome in their method is equal to n. Consequently, for large n, the chromosome length becomes large, and hence the performance of the GA degrades.

4.4. Pixel classification of IRS image of Mumbai

In this section, the utility of the KGA-clustering for partitioning different landcover regions in satellite images is demonstrated. Note that satellite images usually have a large number of overlapping classes and hence the clustering problem in such space becomes quite difficult. This data has been used earlier for classifying the pixels into different categories under the supervised framework [18,19].

4.4.1. IRS image of Mumbai

This 512×512 image was obtained from Indian Remote Sensing Satellite (IRS-1A). This is a circular sun-synchronous satellite, rotating around the earth at the rate of 14 orbits per day, at an altitude of 904 km and a repetition cycle of 22 days. This satellite is equipped with two different sensors *LISS-I* and *LISS-II*. *LISS-I* has a resolution of 72.5×72.5 m^2 while *LISS-II* has a resolution of 36.25×36.25 m^2. Data used for this work was obtained from *LISS-II* sensor which has a focal length of 324.4 m and radiometric resolution of 128. We have considered here two bands, namely,

- green band of wavelength 0.52–0.59 μm,
- near infra red band of wavelength 0.77–0.86 μm.

Some important landcovers of Mumbai, as seen more prominently from near infra red band (Fig. 4 shows the image with histogram equalization to make it more prominent), are as follows: the elongated city area is surrounded by the Arabian sea. There is a concrete structure (on the right side top corner) connecting Mumbai to New Mumbai. On the southern part of the city, there are several islands, including the well known *Elephanta islands*. The dockyard is situated on the south eastern part of Mumbai, which can be seen as a set of three finger like structure. On the upper part of the image, towards left, there is a distinct crisscrossed structure. This is the *Santa Cruz airport*.

Fig. 5 provides the output Mumbai image using KGA-clustering algorithm. From our previous knowledge about this data, we expect the image to contain

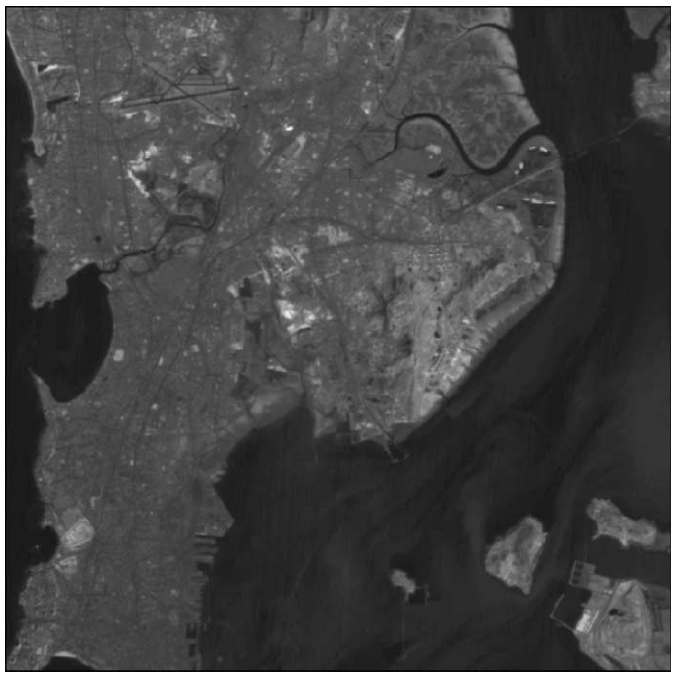

Fig. 4. Near infra red band of the Mumbai image with histogram equalization.

six visually distinct clusters [19]. Hence K was taken to be 6 for this data. As can be seen from the results, most of the landcover categories have been correctly distinguished. For example, the *Santa Cruz* airport, dockyard, bridge connecting Mumbai to New Mumbai, and many other road structures have come out distinctly. Moreover, the predominance of one category of pixels in the southern part of the image conforms to the ground truth; this part is known to be heavily industrialized, and hence the majority of the pixels in this region should belong to the same class of concrete. The *Arabian Sea* has come out as a combination of pixels of two different classes. This is again in conformity with earlier experiments with this data [19], where the sea water was found to be decomposed into two classes, turbid waters 1 and 2, based on the difference of their reflectance properties. (Note that the result presented here does not include any postprocessing operations. One such step that is usually carried out in image processing is the merging of very small segments to one of the neighboring regions. This helps in cleaning up the image by filling up small holes embedded with large regions.) Therefore, from the ground truth available and the performance of KGA-clustering, it can be concluded that it is a useful technique for differentiating the various landcover types present in an image.

Fig. 5. Mumbai image partitioned into six clusters using the KGA-clustering method.

5. Discussion and conclusions

A genetic algorithm-based clustering algorithm, called KGA-clustering, has been developed in this paper. Genetic algorithm has been used to search for the cluster centers such that a given clustering metric, $\mathscr{M}$, is minimized. The knowledge that the centroid of the points belonging to a cluster represents the center of the cluster is incorporated in the chromosome for enhancing the searching capability of the clustering method. Floating point representation of chromosomes has been adopted, since it appears to be a more natural and appropriate form for encoding the cluster centers. The superiority of the KGA-clustering algorithm over K-Means algorithm and another GA-clustering [11] is extensively established for several artificial and real life data sets with dimensions ranging from two to five and number of clusters from two to six. Moreover, the utility of the KGA-clustering algorithm for classifying the pixels of satellite images is also demonstrated for an image of a part of the city of Mumbai. Note that the number of pixels is equal to 262,144 (512×512 image), which is a significantly large data to cluster.

It has been proved in [20] that an Elitist model of GAs will definitely converge to the optimal string as the number of iterations tends to infinity

provided the probability of going from any population to the one containing the optimal string is greater than zero. Since both the GA-based clustering algorithms discussed in this article satisfy this criterion, they will surely provide the optimal clusters as the number of iterations goes to infinity. However, as seen from the results, the speed with which the KGA-clustering algorithm converges to a desired value is significantly higher than that of the GA-clustering algorithm. In real life situations, where the algorithms have to be terminated after finite number of iterations, KGA-clustering algorithm appears to be better applicable than the GA-clustering algorithm [11].

The KGA-clustering algorithm developed in this paper can be applied when the number of clusters is known a priori and they are crisp in nature. As a scope of further research, the algorithm may be modified in order to take care of situations where the number of clusters is unknown as well as when the partitioning is fuzzy. This may necessitate the consideration of variable length chromosomes in GAs to search for the appropriate fuzzy partitions of the data set.

References

[1] M.R. Anderberg, Cluster Analysis for Application, Academic Press, New York, 1973.
[2] A.K. Jain, R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Englewood Cliffs, NJ, 1988.
[3] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, MA, 1974.
[4] S. Guha, R. Rastogi, K. Shim, CURE: an efficient clustering for large databases, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 1998, pp. 73–84.
[5] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining application, in: Proceedings of the ACM SIGMOD Conference on Management of Data (SIGMOD '98), 1998, pp. 94–104.
[6] H. Frigui, R. Krishnapuram, A robust competitive clustering algorithm with application in computer vision, IEEE Trans. Patt. Anal. Machine Intell. 21 (1) (1999) 450–465.
[7] A.K. Jain, P. Duin, M. Jianchang, Statistical pattern recognition: a review, IEEE Trans. Patt. Anal. Machine Intell. 22 (1) (2000) 4–37.
[8] D.E. Goldberg, Genetic Algorithms: Search, Optimization and Machine Learning, Addison-Wesley, New York, 1989.
[9] T.E. Davis, J.C. Principe, A simulated annealing-like convergence theory for the simple genetic algorithm, in: R.K. Belew, J.B. Booker (Eds.), Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, 1991, pp. 174–181.
[10] M. Mitchell, Introduction to Genetic Algorithms, MIT Press, Ann Arbor, MI, 1996.
[11] C.A. Murthy, N. Chowdhury, In search of optimal clusters using genetic algorithms, Pattern Recog. Lett. 17 (1996) 825–832.
[12] J.J. Grefenstette, Incorporating problem specific information in genetic algorithms, in: L. Davis (Ed.), Genetic Algorithms and Simulated Annealing, Pitman, London, 1987, pp. 42–60.
[13] S.Z. Selim, M.A. Ismail, K-means type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Trans. Pattern Anal. Mach. Inteli 6 (1984) 81–87.

[14] H. Spath, Cluster Analysis Algorithms, Ellis Horwood, Chichester, UK, 1989.
[15] S.K. Pal, D.D. Majumder, Fuzzy sets and decision making approaches in vowel and speaker recognition, IEEE Trans. Syst. Man Cybern. SMC-7 (1977) 625–629.
[16] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugen. 3 (1936) 179–188.
[17] R.A. Johnson, D.W. Wichern, Applied Multivariate Statistical Analysis, Prentice-Hall, Englewood Cliffs, NJ, 1982.
[18] S. Bandyopadhyay, C.A. Murthy, S.K. Pal, Supervised pattern classification with surface fitting with genetic algorithms, Proc. Indian Natl. Sci. Acad., Part A 67 (2) (2001) 295–314.
[19] S.K. Pal, S. Bandyopadhyay, C.A. Murthy, Genetic classifiers for remotely sensed images: comparison with standard methods, Int. J. Remote Sensing (2001) 2545–2569.
[20] D. Bhandari, C.A. Murthy, S.K. Pal, Genetic algorithm with elitist model and its convergence, Int. J. Pattern Recog. Art. Intell 10 (1996) 731–747.