Pattern Recognition 42 (2009) 1210 – 1222

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A genetic algorithm with gene rearrangement for K-means clustering

Dong-Xia Chang[a,*], Xian-Da Zhang[a], Chang-Wen Zheng[b]

[a]Tsinghua National Laboratory for Information Science and Technology, State Key Laboratory on Intelligent Technology and Systems, Department of Automation, Tsinghua University, Beijing 100084, China
[b]National Key Lab of Integrated Information System Technology, Institute of Software, Chinese Academy of Sciences, Beijing 100080, China

ARTICLE INFO

Article history:
Received 13 November 2007
Received in revised form 3 November 2008
Accepted 7 November 2008

Keywords:
Clustering
Evolutionary computation
Genetic algorithms
K-means algorithm
Remote sensing image

ABSTRACT

In this paper, a new clustering algorithm based on genetic algorithm (GA) with gene rearrangement (GAGR) is proposed, which in application may effectively remove the degeneracy for the purpose of a more efficient search. A new crossover operator that exploits a measure of similarity between chromosomes in a population is also presented. Adaptive probabilities of crossover and mutation are employed to prevent the convergence of the GAGR to a local optimum. Using the real-world data sets, we compare the performance of our GAGR clustering algorithm with K-means algorithm and other GA methods. An application of the GAGR clustering algorithm in unsupervised classification of multispectral remote sensing images is also provided. Experiment results demonstrate that the GAGR clustering algorithm has high performance, effectiveness and flexibility.

1. Introduction

Clustering [1–5] is an important unsupervised learning technique where a set of patterns, usually vectors in a multidimensional space, is used for identifying groups (clusters) of similar characteristics. More specifically, in clustering, a set of patterns is grouped into clusters in such a way that patterns in the same cluster are similar in some sense and differentiate from those of other clusters in the same sense. Clustering has been applied in a wide variety of fields such as machine learning, pattern recognition, web mining, image segmentation [1].

Traditional classifications of clustering algorithms primarily distinguish between hierarchical and partitional [1,2]. Here, we will use a different categorization that is based on the clustering criterion adopted by the algorithm [6]. In this sense, existing clustering algorithms can be broadly divided into the following four classes.

The first class employs a local concept of clustering based on the idea that neighboring data points should share the same cluster. Algorithms implementing this principle are density-based methods [7,8], nearest neighbor methods [9] and methods like single link agglomerative clustering [10]. These methods are well suited to detect clusters of arbitrary shapes; however, they are not robust when there is little spatial separation between the clusters.

The second class methods are generally implemented by keeping intraclusters variation (i.e., variation between same-cluster data points or between data points and cluster representatives) small. This category includes algorithms like K-means [1–3], average link agglomerative clustering [10], learning-network clustering [11–15], and model-based clustering approaches [16,17]. These methods tend to be very effective for spherical and well-separated clusters, but they may fail for more complicated cluster structures.

The third class performs simultaneous row–column clustering. Typical examples of this kind are biclustering algorithms [18–20]. The goal of these techniques is to identify subgroups of rows and subgroups of columns, by performing simultaneous clustering of both rows and columns of the data matrix, instead of clustering these two dimensions separately. Therefore, biclustering techniques produce local models, whereas clustering approaches compute global models. The clusters identified by these algorithms are not mutually exclusive or exhaustive. A data point may belongs to no cluster or one or more clusters.

The fourth optimizes several validity measures that can capture the different characteristics of the data set. This kind of algorithms can be subdivided into clustering ensembles [21–23], which combine the resulting solutions into a single one with better quality, and multiobjective clustering methods [18,24,25], which provide an estimate of the quality of all individual clusteringsolutions and determine a set of potentially promising clustering solutions. Though often more robust and yield higher quality results than individual clustering methods, these algorithms are not without drawbacks. Clustering ensembles operate with homogenous objective functions. Therefore, good clustering

* Corresponding author. Tel.: +86 10 62794138.
E-mail address: chang_dongxia@hotmail.com (D.-X. Chang).

0031-3203/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2008.11.006

results may become diluted by weak results in an ensemble. For the multiobjective approaches, the construction of the promising solutions set is a difficult conceptual problem, since clustering algorithms often are not accompanied by a measure of the goodness of the detected clusters. Each clustering result should be judged not only by the clustering algorithm that generated it, but also by an external assessment criteria.

Among the second class methods, the K-means algorithm is one of the more widely used algorithms. However, it is well known that K-means algorithm is sensitive to the initial cluster centers [3] and easy to get stuck at the local optimal solutions [26]. Moreover, when the number of data points is large, it takes enormous time to find the global optimal solution [27]. In order to improve the performance of the K-means algorithm, a variety of methods have been proposed [28–30]. In this paper, we attempt to integrate a K-means algorithm with a genetic algorithm (GA) to achieve a better performance.

GAs [31–33], an imitation of natural selection and survival of the fittest, have been proved to be an efficient way in dealing with the optimization problem. In the past years, several clustering algorithms based on GA have been developed. These algorithms use different representations for the clustering solutions. One kind methods uses a straightforward encoding, in which the chromosome is encoded as a string of length n, where n is the number of data points, and the element of the chromosome denotes the cluster number that data point belongs to, such as is used in Refs. [34,35]. This approach does not reduce the size of the search space and searching the optimal solution can be onerous when the data points proliferate. It is for this reason that some researchers opt to use a relatively indirect approach where the chromosome encodes the centers of the clusters, and each data is subsequently assigned to the closest cluster center [36–41]. Tseng and Yang [42] proposes a GA for the clustering problem that is suitable for clustering the data with compact spherical clusters. This algorithm can automatically evolve the number of clusters. But the number of clusters obtained is influenced by some parameters used by the algorithm. In Refs. [43,44], a hyper-quadtree is employed to denote a set of centers. The weakness of this approach is that the need to establish the sets of centers occupying the same region of space can be very time-consuming and prone to bad solutions when such sets are inappropriately selected. Yet, other researchers are seeking to the hybridization between GAs and K-means with interest in using GAs to feature selection for clustering. In Refs. [45,46], GAs are used to evolve the features serving as the input for the K-means algorithm. The clustering solutions of K-mean algorithm are then evaluated and the resulting objective function values are fed back to the GAs. Two coevolutionary algorithms are used for the feature weighting in Ref. [47].

Many of the clustering algorithms based on GAs described above suffer from degeneracy. According to Radcliffe et al. [48], degeneracy occurs when multiple chromosomes represent the same solution. Degeneracy can lead to inefficient coverage of the search space as the same configurations of clusters are repeatedly explored. Furthermore, it is shown that representation with less degeneracy results in more efficient GAs with respect to clustering problems in Ref. [49]. In this paper, a genetic algorithm with gene rearrangement (GAGR) is introduced to enhance the performance of clustering. In GAGR, the degeneracy of chromosome is effectively removed, which makes the evolution process converge fast. Furthermore, a distance measuring the real-valued chromosome is introduced to define a new crossover operator called path-based crossover, which builds a path between two parent chromosomes.

The remainder of the paper is organized as follows. Section 2 provides some definitions necessary for our approach. Then a detail of our GAGR clustering algorithm is presented in Section 3. The ability of the GAGR clustering algorithm to avoid the degeneracy in the evolution is demonstrated in Section 4. The experimental results are given in Section 5. Finally, Section 6 offers conclusions.

2. Preliminaries

In this section, some definitions needed in the next section are given. First, a distance measure for two vectors is defined. This is used in the description of the path-based crossover. Then the definition of gene rearrangement is presented.

Definition 1. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two vectors in the N-dimensional space $\mathbf{R}^N$. Then we say $\mathbf{x}_1 \leqslant \mathbf{x}_2$ if $\mathbf{x}_1(i) \leqslant \mathbf{x}_2(i)$, $1 \leqslant i \leqslant N$ and $\mathbf{x}_1 < \mathbf{x}_2$ if $\mathbf{x}_1(i) < \mathbf{x}_2(i)$, $1 \leqslant i \leqslant N$.

Definition 2. Let $\mathbf{x}_1, \mathbf{x}_2$ be two vectors in $\mathbf{R}^N$. The distance between $\mathbf{x}_1$ and $\mathbf{x}_2$ is defined as

$$Dist(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i=1}^{N} d_i, \tag{1}$$

where $d_i = 1$ if $\mathbf{x}_1(i) \neq \mathbf{x}_2(i)$ and $d_i = 0$ if $\mathbf{x}_1(i) = \mathbf{x}_2(i)$.

Definition 3. Define a new vector $\overline{\mathbf{x}_1\mathbf{x}_2}$, where

$$\overline{\mathbf{x}_1\mathbf{x}_2}(i) = \max\{\mathbf{x}_1(i), \mathbf{x}_2(i)\}. \tag{2}$$

Clearly, $\mathbf{x}_1 \leqslant \overline{\mathbf{x}_1\mathbf{x}_2}$ and $\mathbf{x}_2 \leqslant \overline{\mathbf{x}_1\mathbf{x}_2}$.

Definition 4. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two vectors in $\mathbf{R}^N$ and $\mathbf{x}_1 \neq \mathbf{x}_2$. If $\mathbf{x}_1(i) < \mathbf{x}_2(i)$ for some i, then let j be the smallest integer such that $\mathbf{x}_1(j) < \mathbf{x}_2(j)$, and define the function

$$(\mathbf{x}_1 \uparrow \mathbf{x}_2)(i) = \begin{cases} \mathbf{x}_2(i) & \text{if } i = j, \\ \mathbf{x}_1(i) & \text{otherwise.} \end{cases} \tag{3}$$

If $\mathbf{x}_1(t) > \mathbf{x}_2(t)$ for some t, then let k be the largest integer such that $\mathbf{x}_1(k) > \mathbf{x}_2(k)$, and define the function

$$(\mathbf{x}_1 \downarrow \mathbf{x}_2)(i) = \begin{cases} \mathbf{x}_2(i) & \text{if } i = k, \\ \mathbf{x}_1(i) & \text{otherwise.} \end{cases} \tag{4}$$

In the following, we will give an example for the above definitions. If $\mathbf{x}_1$ and $\mathbf{x}_2$ represent two cluster results of a two-dimensional data set which has three centers, then $\overline{\mathbf{x}_1\mathbf{x}_2}$, $(\mathbf{x}_1 \uparrow \mathbf{x}_2)$ and $(\mathbf{x}_1 \downarrow \mathbf{x}_2)$ are given in Table 1.

Definition 5. Let $\mathbf{x}$ and $\mathbf{y}$ be two vectors in $\mathbf{R}^{KN}$, and can be written as

$$\begin{aligned} \mathbf{x} &= [x_{11}, x_{12}, \ldots, x_{1N}, x_{21}, x_{22}, \ldots, x_{2N}, \ldots, x_{K1}, x_{K2}, \ldots, x_{KN}] \\ &= [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K], \\ \mathbf{y} &= [y_{11}, y_{12}, \ldots, y_{1N}, y_{21}, y_{22}, \ldots, y_{2N}, \ldots, y_{K1}, y_{K2}, \ldots, y_{KN}] \\ &= [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K], \end{aligned}$$

Table 1
Example of above definitions

Function	Example
$\mathbf{x}_1$	[0.1, 0.5, 1.2, 0.3, 0.6, 1.8]
$\mathbf{x}_2$	[0.3, 0.2, 0.9, 0.5, 0.4, 1.9]
$\overline{\mathbf{x}_1\mathbf{x}_2}$	[0.3, 0.5, 1.2, 0.5, 0.6, 1.9]
$\mathbf{x}_1 \uparrow \mathbf{x}_2$	[0.3, 0.5, 1.2, 0.3, 0.6, 1.8]
$\mathbf{x}_1 \downarrow \mathbf{x}_2$	[0.1, 0.5, 1.2, 0.3, 0.4, 1.8]

where $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{iN}]$, $\mathbf{y}_i = [y_{i1}, y_{i2}, \ldots, y_{iN}]$ and $\mathbf{x}$ is called as referenced vector. Let $P_1 = \{1, 2, \ldots, K\}$ and $P_2 = \emptyset$:

```
for i = 1 to K do
        k = arg min_{j∈P1, j∉P2} ||y_i − x_j||^2
        P1 = P1\k
        P2(i) = k
end
```

then the elements of P_2 will be a permutation of that of P_1, and $\mathbf{y}$ will be rearranged according to P_2, i.e.,

$$\mathbf{y}'_k = \mathbf{y}_{P_2(k)}.$$

This implies that the elements of $\mathbf{y}'$ are rearrangement of that of $\mathbf{y}$.

In our algorithm, the chromosome which suffers from this transformation is called chromosome with gene rearrangement.

Proposition 1. *Let* $\mathbf{x}_1$ *and* $\mathbf{x}_2$ *be two vectors in* $\mathbf{R}^N$. *If* $\mathbf{x}_1 \neq \mathbf{x}_2$ *and* $\mathbf{x}_1(i) < \mathbf{x}_2(i)$ *for some* i, *then* $\mathbf{x}_1 \uparrow \mathbf{x}_2$ *exists and*

$$Dist((\mathbf{x}_1 \uparrow \mathbf{x}_2), \mathbf{x}_2) = Dist(\mathbf{x}_1, \mathbf{x}_2) - 1.$$

The proof of this proposition is simple and may not deserve further space.

3. The GAGR clustering algorithm

3.1. Chromosome representation

For any GA, a chromosome representation is needed to describe each chromosome in the population. The representation method determines how the problem is structured in the algorithm and the genetic operators that are used. Each chromosome is made up of a sequence of genes from certain alphabet which can consist of binary digits (0 and 1), floating-point numbers, integers, symbols (i.e., A, B, C, D), etc. In early GAs, the binary digit was used. It has been shown that more natural representations can get more efficient and better solutions. Michalewicz [32] has performed extensive experiments comparing real-valued and binary GAs which indicates that the real-valued GA is more efficient in terms of CPU time. Thus a real-valued representation is utilized to describe the chromosome in this paper.

In this representation each of the centers of the clusters is encoded by the chromosome in the same way as that of [36]. Specifically, each chromosome is described by a sequence of $M = N * K$ real-valued numbers where N is the dimension of the feature space, and K is the number of clusters. That is to say, the chromosome of the algorithm is written as

$$\mathbf{M} = [m_{11}, m_{12}, \ldots, m_{1N}, m_{21}, m_{22}, \ldots, m_{2N}, \ldots, m_{K1}, m_{K2}, \ldots, m_{KN}], \tag{5}$$

where the first N values represent the first cluster center, the second N values represent the second center, and so forth.

3.2. Population initialization

In GAGR clustering algorithm, an initial population of size P can be randomly generated and K data points randomly chosen from the data set but on the condition that there are no identical points to form a chromosome, presenting the K cluster centers. This process is repeated until P chromosomes are generated. Only valid strings (i.e., those that have at least one data point in each cluster) are considered to be included in the initial population.

After the population initialization, each data point is assigned to the cluster with closest cluster center using the following equation:

$$\mathbf{x}_i \in C_j \leftrightarrow \|\mathbf{x}_i - \mathbf{m}_j\| = \min_k \|\mathbf{x}_i - \mathbf{m}_k\|, \quad k = 1, 2, \ldots, K, \tag{6}$$

where $\mathbf{m}_k$ is the center of the kth cluster.

3.3. Fitness function

The fitness function is used to define a fitness value to each candidate solution. A common clustering criterion or quality indicator is the sum of squared error (SSE) measure, defined as

$$\begin{aligned} SSE &= \sum_{C_i} \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{m}_i)^{\mathrm{T}} (\mathbf{x} - \mathbf{m}_i) \\ &= \sum_{C_i} \sum_{\mathbf{x} \in C_i} \|\mathbf{x} - \mathbf{m}_i\|^2, \end{aligned} \tag{7}$$

where $\mathbf{x} \in C_i$ is a data point assigned to that cluster. This measure computes the cumulative distance of each pattern from its cluster center of each cluster individually, and then sums those measures over all clusters. If this measure is small, then the distances from patterns to cluster centers are all small and the clustering would be regarded favorably. It is interesting to note that SSE has a theoretical minimum of zero, which corresponds to all clusters containing only a single data point.

Then the fitness function of the chromosome is defined as the inverse of SSE, i.e.,

$$f = \frac{1}{SSE}. \tag{8}$$

This fitness function will be maximized during the evolutionary process and lead to minimization of the SSE.

3.4. Evolutionary operators

3.4.1. Crossover

The main goal of the crossover operator is to create diversified and potentially promising new chromosomes. It combines the features of two parent chromosomes to form two offspring by swapping corresponding segments of the parents. The intuition behind the applicability of the crossover operator is information exchange between different potential solutions. In this algorithm, two crossover operators are used: path-based crossover and the heuristic crossover. The crossover probability is selected adaptively as in Ref. [50]. Let $f_{\max}$ be the maximum fitness value of the current population, $\bar{f}$ be the average fitness value of the population and f' be the larger of the fitness of the individual to be crossed. Then the probability of crossover p_c is calculated as

$$\begin{aligned} p_c &= k_1 \times \frac{f_{\max} - f'}{f_{\max} - \bar{f}} \quad \text{if } f' > \bar{f}, \\ p_c &= k_3 \quad \text{if } f' \leqslant \bar{f}, \end{aligned} \tag{9}$$

where the values of k_1 and k_3 are equal to 1 [50]. Clearly, when $f_{\max} = \bar{f}$, then $f' = f_{\max}$ and p_c will be equal to k_3. The value of p_c increases when the individual is quite poor. In contrast when the individual is a good solution, p_c will be low so as to reduce the likelihood of disrupting a good solution by crossover.

In the following, the definitions presented in Section 2 are used to define a path-based crossover operator. Let $\mathbf{M}_1$ and $\mathbf{M}_2$ be two individuals in one generation. If $\mathbf{M}_1 \neq \overline{\mathbf{M}_1\mathbf{M}_2}$, then $(\mathbf{M}_1 \uparrow \overline{\mathbf{M}_1\mathbf{M}_2})$ exists. From the definition of $* \uparrow *$, it can be concluded that $\mathbf{M}_1 \leqslant (\mathbf{M}_1 \uparrow \overline{\mathbf{M}_1\mathbf{M}_2})$. By Proposition 1, if $\mathbf{M}_1 \neq \mathbf{M}_2$ and $\mathbf{M}_1(i) < \mathbf{M}_2(i)$ for some i, then $(\mathbf{M}_1 \uparrow \overline{\mathbf{M}_1\mathbf{M}_2})$ exists and it is closer (in terms of the distance defined

by Definition 2) to $\overline{\mathbf{M}_1\mathbf{M}_2}$ than $\mathbf{M}_1$ is. Therefore, we can construct a series of individuals by iterative using Definition 4. Through the iteration, a finite sequence $\mathbf{M}_{11}=\mathbf{M}_1, \mathbf{M}_{12}=(\mathbf{M}_{11}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2}),\ldots,\mathbf{M}_{1(n-1)}=(\mathbf{M}_{1(n-2)}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2})$ and $\mathbf{M}_{1n}=(\mathbf{M}_{1(n-1)}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2})=\overline{\mathbf{M}_1\mathbf{M}_2}$ is obtained. This sequence can be regarded as a path from $\mathbf{M}_1$ to $\overline{\mathbf{M}_1\mathbf{M}_2}$. Similarly, we can define a path from $\overline{\mathbf{M}_1\mathbf{M}_2}$ to $\mathbf{M}_2$. Let $\mathbf{M}_{21}=\overline{\mathbf{M}_1\mathbf{M}_2}$. Using the definition of $(\mathbf{M}_1\downarrow\mathbf{M}_2)$, sequence $\mathbf{M}_{21},\mathbf{M}_{22},\ldots,\mathbf{M}_{2m}$ can be gotten where $\mathbf{M}_{21}=\overline{\mathbf{M}_1\mathbf{M}_2}$, $\mathbf{M}_{2i}=(\mathbf{M}_{2(i-1)}\downarrow\mathbf{M}_2)$, and $\mathbf{M}_{2m}=\mathbf{M}_2$. So a path from $\mathbf{M}_1$ to $\mathbf{M}_2$ can be acquired by combining the two paths discussed above.

The path-based crossover operator works by building a path between two parent chromosomes. The children are two points selected from this path. The selection method may be a random selection, roulette wheel selection, tournament and so on. Here, we select the best chromosome in term of the fitness and a random one.

In order to explain the path between two chromosomes more clearly, an example is given. Let

$$\mathbf{M}_1=[0.1,0.5,1.2,0.3,0.6,1.8]$$

and

$$\mathbf{M}_2=[0.3,0.2,0.9,0.5,0.4,1.9]$$

be two clustering results of a two-dimensional data set which has three centers, then

$$\overline{\mathbf{M}_1\mathbf{M}_2}=[0.3,0.5,1.2,0.5,0.6,1.9]$$

and the path between $\mathbf{M}_1$ and $\mathbf{M}_2$ is

$$\begin{aligned}
&\mathbf{M}_{11}=\mathbf{M}_1=[0.1,0.5,1.2,0.3,0.6,1.8],\\
&\mathbf{M}_{12}=(\mathbf{M}_{11}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2})=[0.3,0.5,1.2,0.3,0.6,1.8],\\
&\mathbf{M}_{13}=(\mathbf{M}_{12}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2})=[0.3,0.5,1.2,0.5,0.6,1.8],\\
&\mathbf{M}_{14}=(\mathbf{M}_{13}\uparrow\overline{\mathbf{M}_1\mathbf{M}_2})=[0.3,0.5,1.2,0.5,0.6,1.9]=\overline{\mathbf{M}_1\mathbf{M}_2}=\mathbf{M}_{21},\\
&\mathbf{M}_{22}=(\mathbf{M}_{21}\downarrow\mathbf{M}_2)=[0.3,0.5,1.2,0.5,0.4,1.9],\\
&\mathbf{M}_{23}=(\mathbf{M}_{22}\downarrow\mathbf{M}_2)=[0.3,0.5,0.9,0.5,0.4,1.9],\\
&\mathbf{M}_{24}=(\mathbf{M}_{23}\downarrow\mathbf{M}_2)=[0.3,0.2,0.9,0.5,0.4,1.9]=\mathbf{M}_2.
\end{aligned}$$

There also exits a problem in the path-based crossover defined above. When the parents are very close, the path-based crossover will make the search process invalid. We say two parents are close, if the distance (defined by Definition 2) between them is smaller than a small integer (e.g., 2). At this circumstance, other crossover should be used to produce new chromosomes. Here we use the heuristic crossover which utilizes the fitness information. Let $\mathbf{x}$ and $\mathbf{y}$ be two chromosomes to be crossed, then

$$\begin{aligned}
&\mathbf{x}'=\mathbf{x}+r(\mathbf{x}-\mathbf{y}),\\
&\mathbf{y}'=\mathbf{x},
\end{aligned}\tag{10}$$

where $r=U(0,1)$, $U(0,1)$ is a uniform distribution on interval $[0,1]$ and $\mathbf{x}$ is better than $\mathbf{y}$ in term of the fitness.

3.4.2. Mutation

Mutation arbitrarily alters one or more genes of a selected chromosome, by a random change with a probability equal to the mutation rate. The intuition behind the mutation operator is the introduction of some extra variability into the population. Here, the mutation probability is also selected adaptively as [50]. The expression for mutation probability p_m is given below

$$\begin{aligned}
&p_m=k_2\times\frac{f_{\max}-f}{f_{\max}-\bar{f}} \quad \text{if } f>\bar{f},\\
&p_m=k_4 \quad \text{if } f\leqslant\bar{f},
\end{aligned}\tag{11}$$

where k_2 and k_4 are equal to 0.5, $f_{\max}$ and $\bar{f}$ are the same as defined above, and f is the fitness of the chromosome under mutation. From the expressions of p_c and p_m, it is seen that p_c and p_m will get lower values for high fitness solutions and get higher values for low fitness solutions. While the high fitness solutions aid in the convergence of the GA, the low fitness solutions prevent the GA from getting stuck at a local optimum. In order to prevent the GA from getting stuck at a local optimum, the solutions with subaverage fitnesses are employed to search the search space for the region containing the global optimum. Such solutions are disrupted with a high probability ($p_m=0.5$) and new solutions are created. As a result the algorithm will come out of local optimum. But for the solution with the maximum fitness value, p_c and p_m are both zero. The best solution in a population is transferred undisrupted into the next generation. Together with the selection mechanism, this may lead to an exponential growth of the solution in the population and may cause premature convergence. To overcome this problem, a default mutation rate (of 0.001) is introduced for every solution.

The mutation process adopted in this paper is the same as that used in [36] which will be described as below. Let $f_{\min}$ and $f_{\max}$ be the minimum and maximum fitness values in the current population, respectively. For an individual with fitness value f, a number δ in the range $[-R,+R]$ is generated with uniform distribution, where

$$R=\begin{cases}\dfrac{f-f_{\min}}{f_{\max}-f_{\min}}, & f_{\max}>f,\\ 1, & f_{\max}=f.\end{cases}\tag{12}$$

If the minimum and maximum values of the data set along the ith dimension ($i=1,2,\ldots,N$) are $m^i_{\min}$ and $m^i_{\max}$, respectively, then after mutation the ith element of the individual is given by

$$\begin{cases}m^i+\delta\times(m^i_{\max}-m^i), & \delta\geqslant 0,\\ m^i+\delta\times(m^i-m^i_{\min}), & \delta<0.\end{cases}\tag{13}$$

3.5. Description of the algorithm

In our GAGR clustering algorithm, a chromosome representing the cluster centers is used and each chromosome is individually evaluated by using the fitness function described in Section 3.3. In the evolutionary loop, a set of individuals is selected for evolutionary crossover and mutation. A roulette wheel selection of slots is used to implement the selection process. The chance for a chromosome to be selected is proportional to its fitness value.

The probability of evolutionary operator is selected adaptively. The crossover operator transforms two individuals (parents) into two offspring by combining parts from each parent. Herein two kinds of crossover operators are used: path-based and heuristic crossover. The mutation operator operates on a single individual and creates an offspring by mutating that individual (see Section 3.4 for details on evolutionary operators). The newly generated individuals are evaluated on the basis of the fitness function and form the new generation. In every generation, the chromosome with the best fitness value is chosen as the referenced vector. Then other chromosomes suffered from gene degeneracy will be rearranged using Definition 5. Each generation makes use of the elitist strategy [33] by replacing the worst chromosome of current population with the best one seen up to the previous generation. The process terminates after some number of generations either by the user or dynamically by the program itself, where the best chromosome obtained will be taken as the best solution.

The GAGR clustering algorithm is described as follows:

1. Initialize a group of cluster centers with size of P, only valid chromosomes (that have at least one data point in each cluster) are taken into consideration. Each data point of the set is assigned to the cluster with closest cluster center using the Euclidean distance.

2. Evaluate each chromosome and copy the best chromosome say p_{best} of the initial population in a separate location.
3. If the termination condition is not reached, go to Step 4. Otherwise, select the best individual from the population as the best cluster result.
4. Select individuals from the population for crossover and mutation.
5. Apply crossover operator to the selected individuals based on the crossover probability.
6. Apply mutation operator to the selected individuals based on the mutation probability.
7. Evaluate the newly generated candidates.
8. Compare the worst chromosome in the new population with p_{best} in term of their fitness values. If the former is worse than the later, then replace it by p_{best}.
9. Find the best chromosome in the new population and replace p_{best}.
10. For the new population, select the best chromosome as a reference, which other chromosomes might fall into the gene rearrangement if needed.
11. Go back to Step 3.

4. Analysis of the efficiency of GAGR

In fact, the representations used in many clustering problems suffer from degeneracy which can be avoided in the GAGR clustering algorithm by the gene rearrangement. The degeneracy mainly arises from a non-one-to-one correspondence between the representation and the clustering result. For example, $\mathbf{M}_1$=[1.2, 2.1, 3.3, 3.3, 2, 3] represents a clustering result of a two-dimensional data set which has three centers, while $\mathbf{M}_2 = [3.3, 3.3, 2, 3, 1.2, 2.1]$ represents the same clustering result for the same data set. Indeed, any permutation of the centers gives rise to a chromosome that represents an identical clustering result. If the two chromosomes described above are crossed after the second position, then we get offspring [1.2, 2.1, 2, 3, 1.2, 2.1] and [3.3, 3.3, 3.3, 3.3, 2, 3]. These two offspring are invalid since there always exist two genes describing the same cluster. This leads to an inefficient search. But through the gene rearrangement, this kind of degeneracy will be avoided successfully. All the permutation of the cluster centers described by the chromosomes can be transformed into one representation which is the same as the referenced chromosome.

Moreover, degeneracy always exists when the chromosomes are not the same. In the following, we will take an example to describe this problem. Let $\mathbf{M}_1 = [1.1, 1.0, 2.2, 2.0, 3.4, 1.2]$ and $\mathbf{M}_2 = [3.2, 1.4, 1.8, 2.2, 0.5, 0.7]$, respectively, represent two clustering results of a two-dimensional data set with three clusters in one generation. If $\mathbf{M}_1$ is selected as the referenced vector, then $\mathbf{M}_2$ becomes $\mathbf{M}'_2 = [0.5, 0.7, 1.8, 2.2, 3.2, 1.4]$ after the gene rearrangement described by Definition 5. Fig. 1 provides a crossover results of $\mathbf{M}_1$ and $\mathbf{M}_2$ ($\mathbf{M}'_2$). From Fig. 1, we can see the performance of the chromosome marked by triangles in the figure directly obtained from crossing $\mathbf{M}_1$ and $\mathbf{M}_2$ is poor because of some confusion between the two clusters while the one marked by circles in the figure obtained from $\mathbf{M}_1$ and $\mathbf{M}'_2$ is more efficient.

Therefore, there is a one-to-one correspondence between the representation and the clustering result through the gene rearrangement. There is no degeneracy thus makes the algorithm more efficient.

5. Experiments results

For the purpose of testing the performance of the GAGR clustering algorithm, experiments are conducted on both real-world data from the UCI Machine Learning Repository [51] and multispectral remote sensing images, and the results show that GAGR has high performance, effectiveness and flexibility.

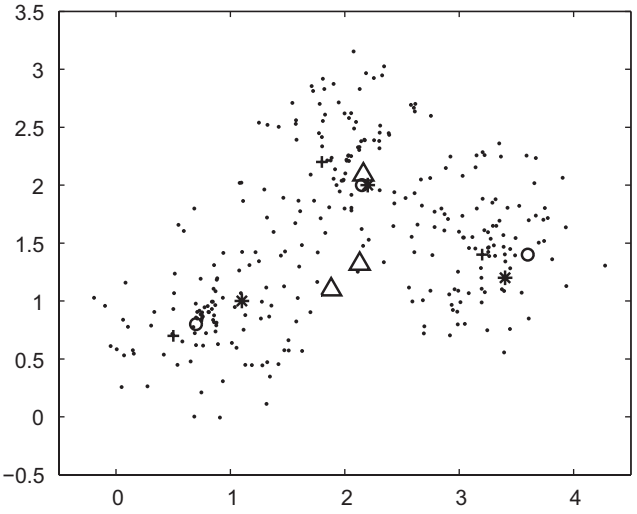

Fig. 1. An example of degeneracy occurs through the crossover. Here, $*$ denotes the chromosome $\mathbf{M}_1$, $+$ denotes the chromosome $\mathbf{M}_2$ ($\mathbf{M}'_2$), $\triangle$ denotes the chromosome obtained by cross $\mathbf{M}_1$ and $\mathbf{M}_2$, $\circ$ denotes the chromosome obtained by cross $\mathbf{M}_1$ and $\mathbf{M}'_2$.

5.1. Experiments on UCI data

In this section, the performances of the GAGR clustering, GA-clustering [34], KGA-clustering [36] and K-means algorithms are compared through the experiments based on the following six real-world data sets that are used:

Iris: Iris data set consists of 150 data points distributed over three clusters. Each cluster has 50 points. This data set represents different categories of irises characterized by four feature values in centimeters: the sepal length, sepal width, petal length and the petal width. This data set has three classes, namely, Setosa, Versicolor and Virginica among which the last two classes have a large amount of overlap while the first class is linearly separable.

Wine: This is the wine recognition data consisting of 178 instances with 13 features resulting from a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wine.

Breast cancer: This 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.

Glass: This is the glass identification data set consisting of 214 data points with nine features (the Id number feature has been moved). The study of classification of types of glass was motivated by criminological investigation. There are six categories in this data set.

Balance: This data set contains 569 data points having four features generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left or be balanced. The attributes are the left weight, the left distance, the right weight, and the right distance.

Liverdisorder: This data set contains 345 instances with six features each. The data have two categories. The first five variables are all blood tests which are thought to be sensitive to liver disorders that might arise from excessive alcohol consumption.

For convenience, we summarize the six sets in Table 2 with the characteristics of the data sets. The three columns show the number of data points M, the number of classes K, and the dimension of the feature space d for each data set.

In the experiments, the population size is taken as 50. The crossover and mutation probabilities for GA-clustering and KGA-clustering algorithm are $p_c = 0.8$ and $p_m = 0.001$, respectively. All the algorithms have the same initialization. The total number of generations is equal to 50 (except for GA-clustering algorithm). Executing it further does not improve the performance. In the experiments, each attribute of the patterns is standardized by dividing the sample range of itself [52]. For comparison purpose, two different performance measures are used: the average SSE of the population vs. the number of iterations and the Rand Index [53].

Table 2
Six UCI data sets used in our experiments

Data set	M	K	d
Iris	150	3	4
Wine	178	3	13
Breast	683	2	9
Glass	214	6	9
Balance	569	3	4
Liverdisorder	345	2	6

The Rand Index [53] measures the agreement of the clustering result with the true cluster structure. It counts the number of pairwise co-assignments of data items between the two partitions. Let n_s be the number of pairs of patterns that are assigned to the same cluster in both the resultant partition and the true cluster structure, and n_d be the number of pairs of patterns that are assigned to different clusters in both the resultant partition and the true cluster structure. The Rand index is defined as

$$RI = \frac{n_s + n_d}{C_n^2} = \frac{n_s + n_d}{n(n-1)/2}, \tag{14}$$

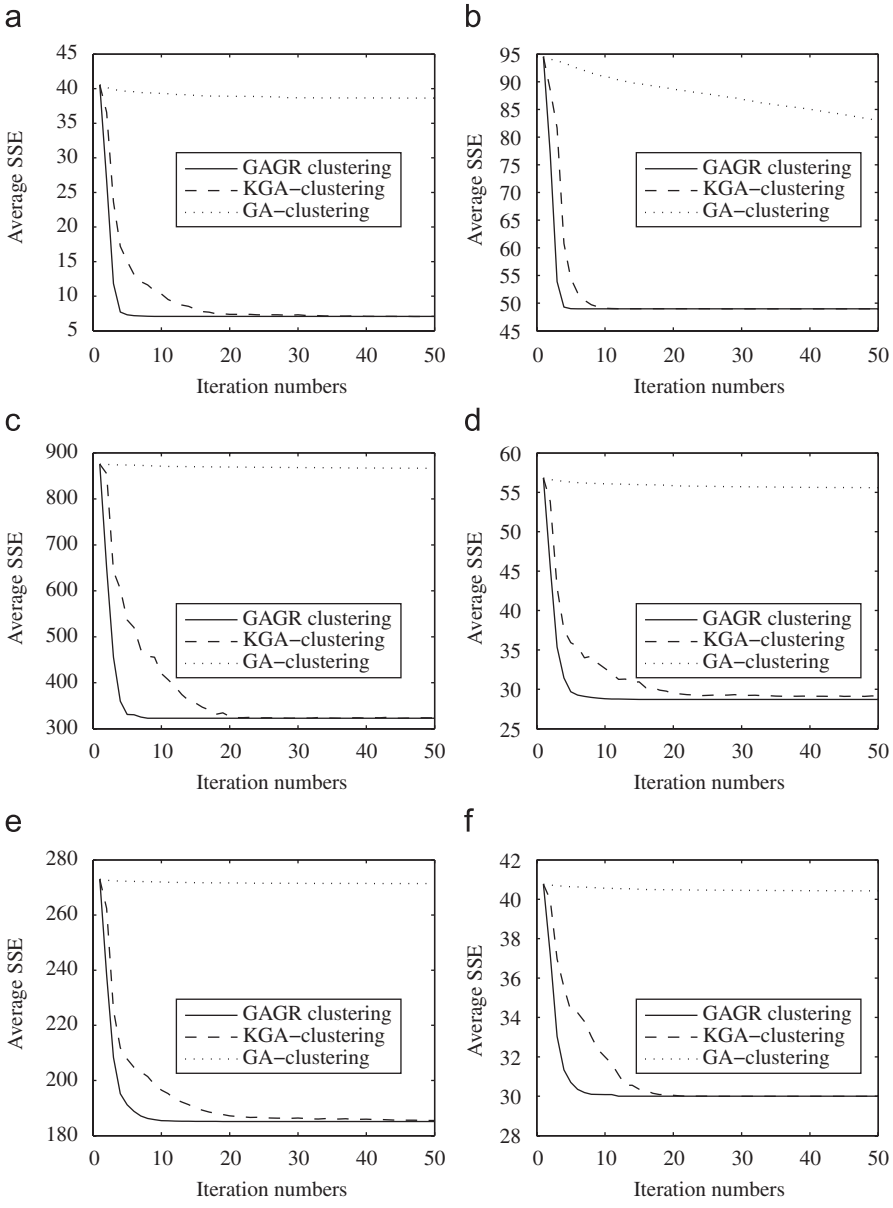

Fig. 2. Clustering results for the six UCI data sets. The figures plot the SSE obtained by the GA-clustering, KGA-clustering and GAGR clustering algorithm and averaged over 20 independent experiments: (a) iris; (b) wine; (c) breast cancer; (d) glass; (e) balance; (f) liverdisorder.

Table 3
The average number of iterations and the computation time to convergence for GA-clustering, KGA-clustering and GAGR clustering algorithms for 20 different runs for the six real-life data sets

Data	GA-clustering		KGA-clustering		GAGR clustering	
	Avg. number of iterations	Computation time (s)	Avg. number of iterations	Computation time (s)	Avg. number of iterations	Computation time (s)
Iris	868.85	45.295	17.60	0.4494	**6.95**	**0.3741**
Wine	821.05	46.478	12.93	0.4960	**6.35**	**0.4011**
Breast	2108.45	235.368	8.12	0.4539	**4.23**	**0.3372**
Glass	534.60	23.867	16.60	0.5969	**7.10**	**0.5613**
Balance	2852.00	230.589	23.57	0.7442	**10.03**	**0.6602**
Liverdisorder	1229.9	71.706	17.730	0.4899	**8.4**	**0.4451**

The bold font are the best value for each data.

where n is the number of elements to be clustered. The Rand Index return values in the interval $[0,1]$ and the optimum score is 1, with higher scores being "better".

The determination of the number of clusters is important in clustering problem. Many methods have been proposed for identifying the number of clusters automatically [15,42,54,55]. In this paper, a simple method discussed in [5] is used to determine the number of clusters. The SSE decreases monotonically with the cluster number. It decreases rapidly until $k=\hat{k}$, and decreases much more slowly thereafter until it reaches zero at $k=n$. Therefore, we examine the plot of the SSE against the number of clusters K, and look for an inflexion of the curve showing that little improvement in the description of the data structure is to be gained above a particular value of K. After determining the number of clusters, experiments are conducted with the determined cluster number.

First, we compare the speed of convergence of the three GA-based algorithms. For each data set we have conducted the experiment 20 independent trials with randomly generated initialization and the average value recorded to account for the stochastic nature of the algorithm. The average SSEs obtained by the three algorithms are shown in Fig. 2 and the characteristic of the computation time in Table 3.

From Fig. 2, it is seen that the GAGR clustering algorithm converges to the desired value in a relatively fewer number of iterations for all the data sets. It is also seen that GAGR clustering algorithm provides some improvement in the SSE over the GA-clustering, and KGA-clustering for some data sets. Table 3 gives the average number of iterations and the computation time (the experiments were implemented on a machine running Windows XP, Intel (R) Xeon (R) CPU, 2.33 GHz) needed for convergence of GA-clustering, KGA-clustering and GAGR clustering algorithm. As seen from Table 3, the GAGR clustering algorithm converges in relatively fewer number of iterations and shorter computation time.

In the following, the Rand Index is employed to compare the performance of the three GA-based algorithms and the K-means. For convenience, the result figures are given to compare the clustering accuracy. Here, we only give the results of the first two data sets as example. These two data sets after dimension reduction by using the PCA [56,57] and the results obtained from the four algorithms are shown in Figs. 3, 4, 5 and 6, respectively. Then, the mean and variance of Rand Index for the six real-world data sets obtained by the four techniques are given in Table 4 which illustrates the Rand Index obtained by GAGR clustering algorithm is always better than that obtained by the other algorithms for the same data sets.

For a more careful comparison among algorithms, the multivariate analysis of variance (MANOVA) technique [58] is used to assess the cluster differences between the actual clusters and those obtained by K-means, KGA-clustering, GA-clustering and GAGR clustering algorithm. MANOVA as a powerful statistical tool provides information on the nature and predictive power of the independent measures. It gauges the group difference between two or more metric dependent variables simultaneously, using a set of categorical non-metric variables. Here, the categorical non-metric variables are the cluster labels. The results are given in Table 5.

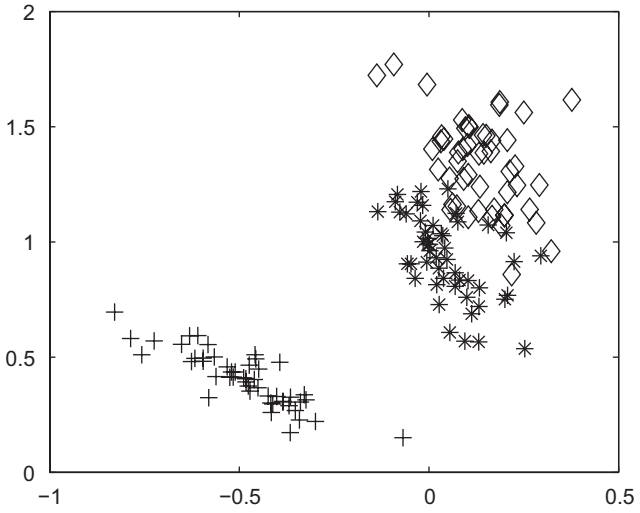

Fig. 3. The two-dimensional iris data after dimension reduction.

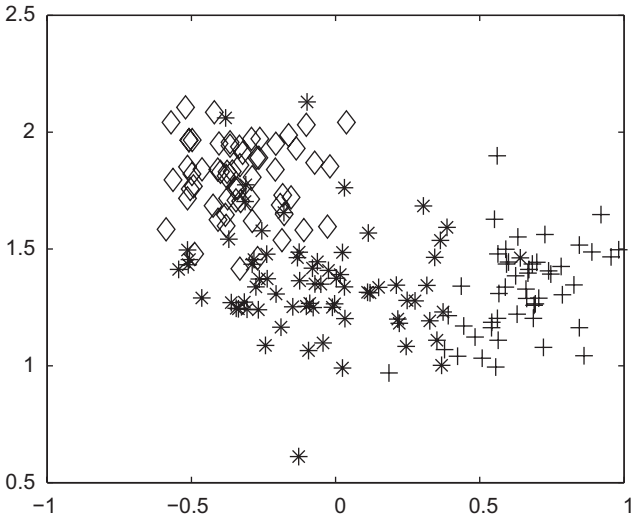

Fig. 4. The two-dimensional wine data after dimension reduction.

In the experiment, MANOVA tests the null hypothesis that the mean of each group is the same dimensional multivariate vector, and that any difference observed in the sample is due to random chance. There are three outputs, d, p and a distance between the group means, in the experiment. If $d=0$, there is no evidence to reject

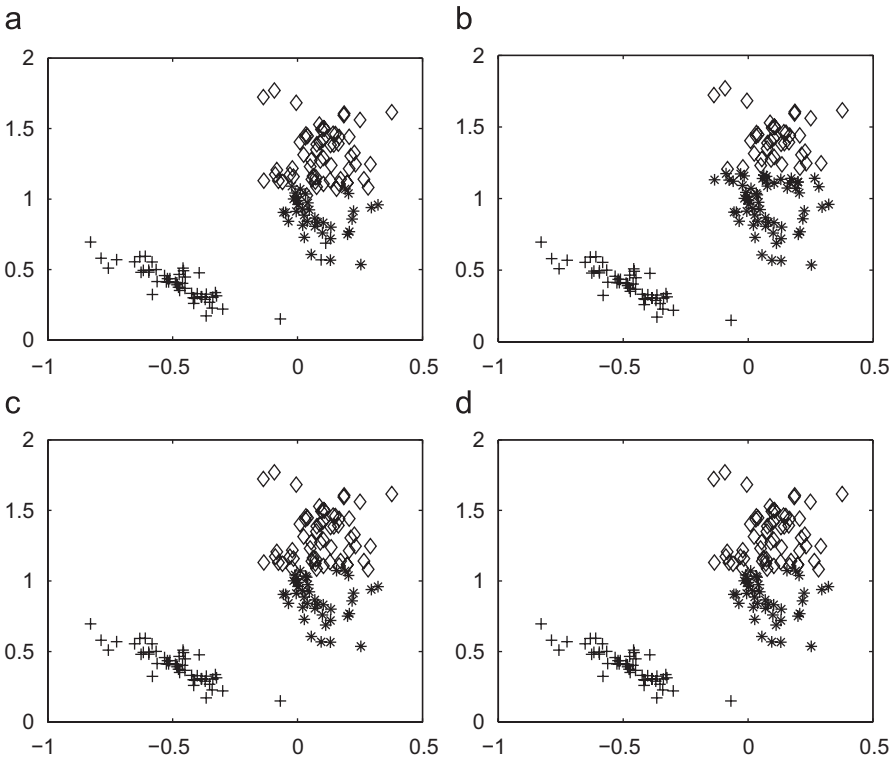

Fig. 5. The cluster results of the iris data using: (a) K-means; (b) GA-clustering; (c) KGA-clustering; (d) GAGR.

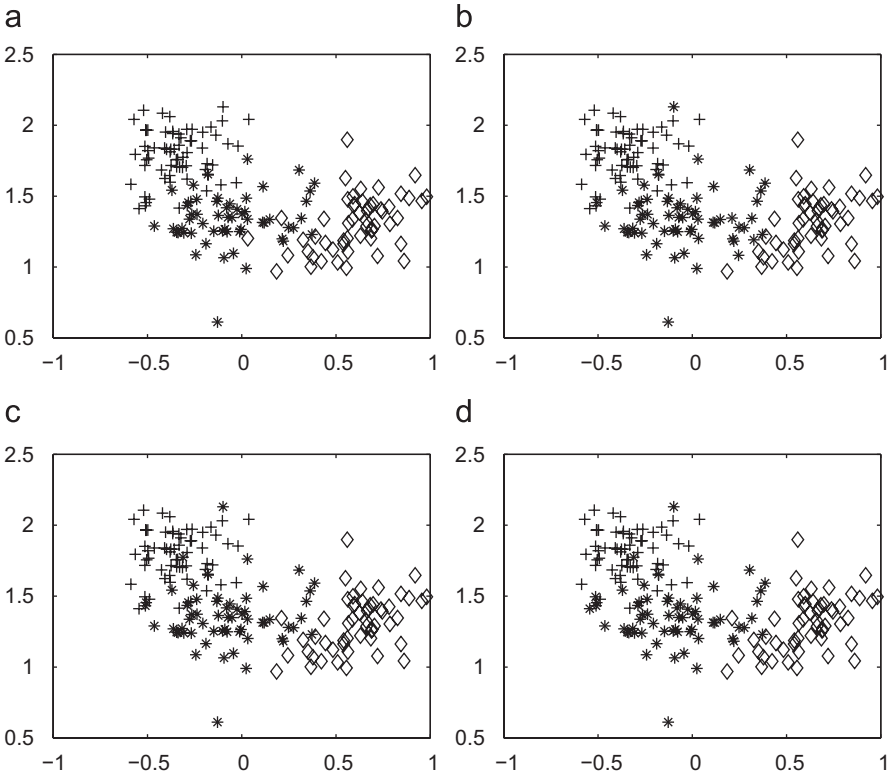

Fig. 6. The cluster results of the wine data using: (a) K-means; (b) GA-clustering; (c) KGA-clustering; (d) GAGR.

Table 4
The maximum, mean and variance, and minimum values of Rand Index obtained by K-means, GA-clustering, KGA-clustering and GAGR algorithms for 20 different runs for the six real-life data sets

Data	K-means	GA-clustering	KGA-clustering	GAGR clustering
Iris				
Max	0.8623	0.8622	0.8737	0.8737
Mean	$0.8292 \pm 4.7873e-4$	$0.8518 \pm 7.2819e-6$	0.8588 ± 0.0013	$\mathbf{0.8711 \pm 1.2207e-5}$
Min	0.7716	0.8564	0.8464	**0.8623**
Wine				
Max	0.9265	0.9415	0.9415	0.9415
Mean	0.8413 ± 0.002	$0.9368 \pm 2.9083e-5$	$0.9319 \pm 6.9322e-5$	$\mathbf{0.9388 \pm 1.1011e-5}$
Min	0.7339	0.9246	0.9120	**0.9349**
Breast				
Max	0.9328	0.9486	0.9486	0.9486
Mean	$0.9279 \pm 5.2901e-5$	$0.9482 \pm 1.0325e-6$	$0.9482 \pm 1.0325e-6$	$\mathbf{0.9486 \pm 5.1899e-32}$
Min	0.9121	0.9458	0.9458	**0.9486**
Glass				
Max	0.5107	0.5308	0.5367	0.5367
Mean	0.3807 ± 0.0059	0.3878 ± 0.0052	0.3966 ± 0.0062	$\mathbf{0.4047 \pm 0.0055}$
Min	0.2556	0.2759	0.2683	**0.2736**
Balance				
Max	0.5826	0.5827	0.5834	**0.6075**
Mean	$0.5603 \pm 1.9476e-4$	$0.5642 \pm 3.9543e-4$	$0.5642 \pm 1.4299e-4$	$\mathbf{0.5646 \pm 3.2559e-4}$
Min	0.5215	0.5345	0.5296	**0.5320**
Liverdisorder				
Max	0.5004	0.5014	0.5043	0.5050
Mean	$0.4989 \pm 2.5786e-7$	$0.5004 \pm 2.4456e-7$	$0.5022 \pm 2.5427e-7$	$\mathbf{0.5031 \pm 1.0624e-6}$
Min	0.4984	0.4993	0.5021	0.5021

The bold font are the best value for each data.

Table 5
Result of MANOVA testing by K-means, GA-clustering, KGA-clustering and GAGR clustering algorithms on the data sets, here *gm* stands for Mahalanobis distance and $dataname_i$ denotes the cluster number of the data set

Data	K-means			GA-clustering			KGA-clustering			GAGR clustering		
	d	*p*	*gm*	*d*	*p*	*gm*	*d*	*p*	*gm*	*d*	*p*	*gm*
$Iris_1$	0	1	0	0	1	0	0	1	0	0	1	0
$Iris_2$	0	0.2640	0.2417	0	0.0909	0.3111	0	0.0927	0.3468	0	0.1077	0.2917
$Iris_3$	0	0.0798	0.3143	0	0.2039	0.2830	0	0.0885	0.3374	0	**0.2324**	**0.2696**
$Wine_1$	0	0.9756	0.1699	0	1	0	0	0.9997	0.0714	0	**1**	**0**
$Wine_2$	0	0.9490	0.2165	0	0.9962	0.1090	0	0.9990	0.0584	0	**0.9980**	**0.0949**
$Wine_3$	0	0.8546	0.3282	0	0.9834	0.1955	0	0.9982	0.1268	0	0.9938	0.1599
$Breast_1$	0	0.9594	0.0169	0	0.9594	0.0169	0	0.9690	0.0157	0	0.9594	0.0169
$Breast_2$	0	0.9992	0.0096	0	0.9999	0.0072	0	0.9998	0.0085	0	**0.9999**	**0.0072**

The bold font are the best value for each data.

Fig. 7. The first pseudocolor image of parts of MiYun obtained from Landsat-7 multispectral scanner composite by displaying band 5 as red, band 4 as green, and band 3 as blue.

Fig. 8. The second pseudocolor image of parts of MiYun obtained from Landsat-7 multispectral scanner composite by displaying band 5 as red, band 4 as green, and band 3 as blue.

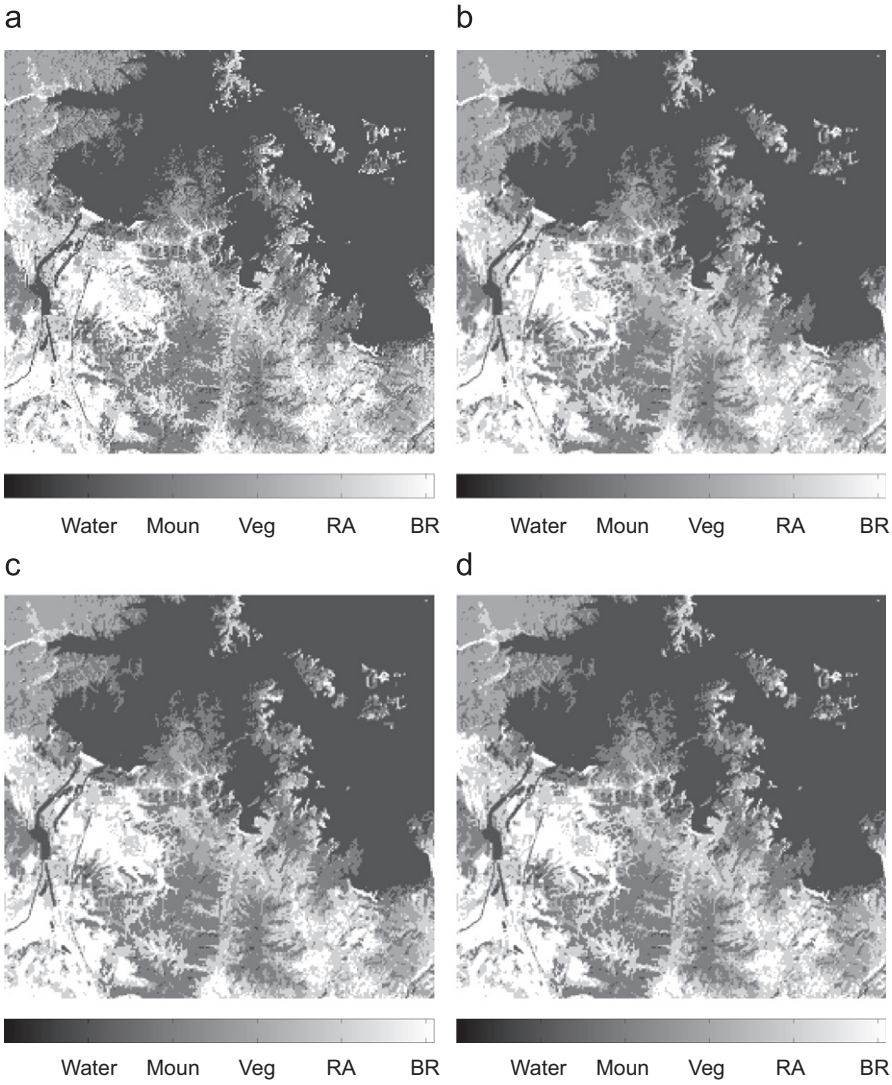

Fig. 9. The clustering results of the remote sensing image shown in Fig. 7 using: (a) K-means; (b) GA-clustering; (c) KGA-clustering; (d) GAGR.

the null hypothesis; while if $d=1$, we can reject the null hypothesis that the means are the same but we cannot reject the hypothesis that the multivariate means lie on the same line. Similarly, if $d=2$, then the multivariate means may lie on the same plane in N-dimensional space, but not on the same line. p is the probability, computed assuming that the null hypothesis is true. The smaller the p is, the stronger is the evidence against the null hypothesis provided by the data. If $p<0.05$, then we will reject the null hypothesis. The gm in Table 5 represents the Mahalanobis distance between each pair of the group means.

It can be seen from Table 5 GAGR is able to find the first cluster correctly ($d=0, p=1, gm=0$) for iris and wine data set. This signifies that the means of data items forming cluster 1 after application of the algorithms and the means of the actual cluster are the same. K-means, GA-clustering and KGA-clustering algorithm only find the first cluster correctly for iris data set. And all the algorithms can not find the other two clusters accurately for the two sets. It is also confirmed by the values of p. Although the values of d obtained are 0 but the result is not significant as the values of p are small. But p value of GAGR is better than p values of all the other algorithms for this two data sets. This is also evident from the fact that the gm value obtained by GAGR is smaller than the gm values obtained by all the other algorithms. For the breast cancer data set, GAGR, GA-clustering and KGA-clustering algorithms perform similarly for both clusters 1 and 2. For all the algorithms we get $d=0$, but p values of GAGR, GA-clustering and KGA-clustering are better than p values of K-means algorithm.

5.2. Experiment on remote sensing image clustering

Remote sensing image analysis is attracting a growing interest in real-world applications. The design of robust and efficient clustering algorithms becomes one of the most important issues addressed by the remote sensing community. In this section, we will apply K-means, GA-clustering, KGA-clustering and GAGR to the clustering of multispectral remote sensing image based on the spectral data of pixels. Although the remote sensing images usually have a large number of overlapping clusters, the experimental results show that the multispectral image can be effectively grouped into several clusters by the proposed method.

In this experiment, the algorithms are used to partition different landcover regions in two remote sensing images. Two 512×512 remote sensing images of different parts of MiYun

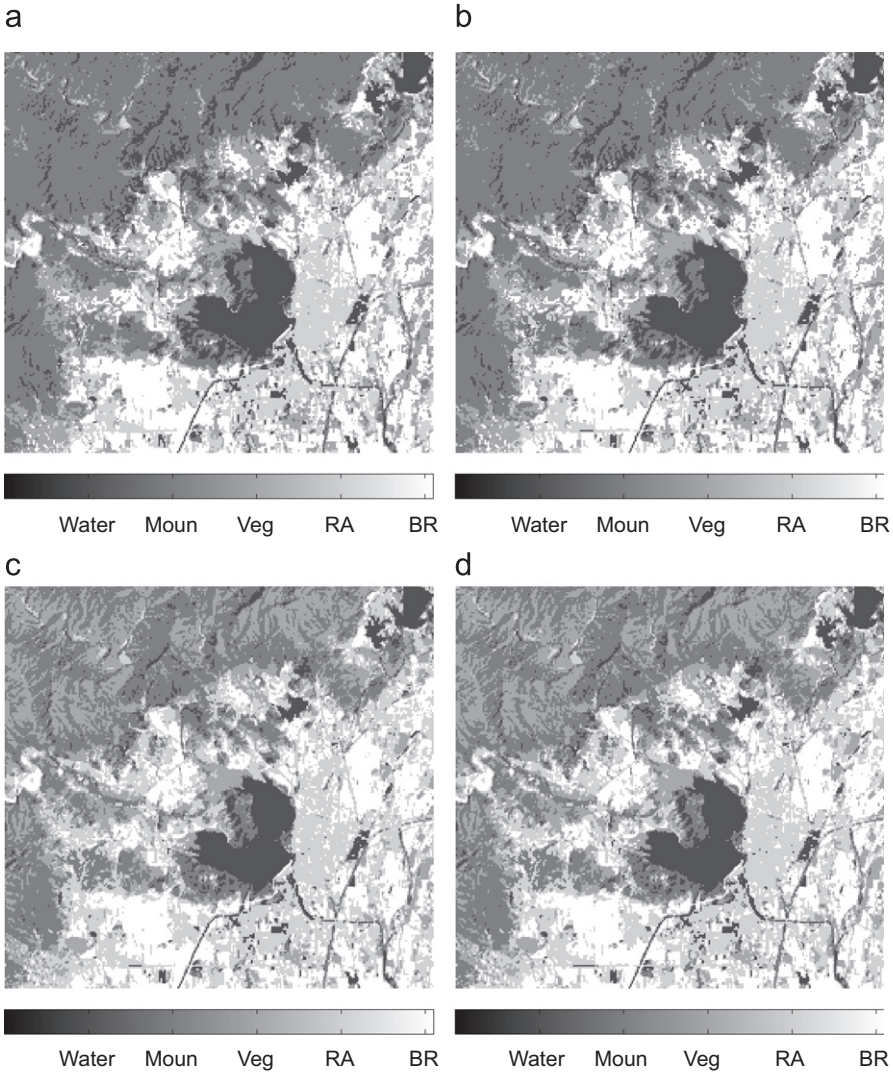

Fig. 10. The clustering results of the remote sensing image shown in Fig. 8 using: (a) K-means; (b) GA-clustering; (c) KGA-clustering; (d) GAGR.

Table 6
The SSE and validity index $\mathscr{I}$ obtained by K-means, KGA-clustering, GA-clustering and GAGR algorithms on the remote sensing images

Image	K-means	GA-clustering	KGA-clustering	GAGR clustering
First				
SSE	6.3014e + 007	5.5711e + 007	5.3013e + 007	**5.2901e + 007**
$\mathscr{I}$	35190.8620	36117.9402	36329.9851	**36477.32**
Second				
SSE	9.4598e + 007	9.3736e + 007	9.2895e + 007	**9.2073e + 007**
$\mathscr{I}$	9088.7062	9117.8326	10640.4371	**10745.6446**

The bold font are the best value for each data.

obtained from Landsat-7 have been chosen. The images considered have three bands in the multispectral mode: band 3—red band, wavelength 0.63–0.69 μm; band 4—near-infrared band, wavelength 0.76–0.94 μm; band 5—shortwave infrared band, wavelength 1.55–1.75 μm. The pseudocolor images are shown in Figs. 7 and 8, respectively.

From the pseudocolor images, it can be seen that the landcovers of the images mainly contains five classes: water, vegetation (Veg), mountain (Moun), residential areas (RA) and blank regions (BR). In the experiment, we expect the four algorithms can partition the remote sensing images into five visually distinct clusters. The number of population is set to 50 and the maximum generation 200. The crossover and mutation probabilities are the same as those used in the first experiment. The clustering results for the two images are shown in Figs. 9 and 10 with gray scale, respectively.

As shown in Figs. 9 and 10, most of the landcover categories have been correctly distinguished by the four algorithms. For example, the mountain, the rivers in the residential areas and many other structures are identified by the clustering algorithm. So we can conclude that GAGR clustering algorithm is an efficient clustering algorithm

for differentiating the various landover types present in the image. But it is difficult to distinguish the difference between the four algorithms by the gray scale results. To evaluate the clustering results more carefully, the SSE and the validity index $\mathscr{I}$ are used.

The validity index $\mathscr{I}$ [59] is used to measure the clustering performance. It has been proposed as a measure of indication the goodness/validity of a cluster solution. It is defined as follows:

$$\mathscr{I}(K)=\left(\frac{1}{K}\times\frac{E_1}{E_K}\times\mathscr{D}_K\right)^p, \tag{15}$$

where $E_K=\sum_{k=1}^{K}\sum_{j=1}^{n}D(x_j,c_k)$ and $\mathscr{D}_K=\max_{i,j=1}^{K}\{D(c_i,c_j)\}$. The power p is used to control the contrast between the different cluster configurations. Here, we let $p=2$ and it is the same as used in Ref. [59]. A larger value of $\mathscr{I}$ index implies a better solution. Note that for computing the index $\mathscr{I}$, knowledge about the true partitioning of the data is not necessary.

The SSE and the $\mathscr{I}$ index values obtained by the four algorithms are presented in Table 6. The values indicate the superiority of the GAGR clustering algorithm which produces a better value of the SSE and the $\mathscr{I}$ index values than those of other algorithms. Through the experiments, it can be concluded that GAGR clustering algorithm is a useful method for remote sensing image clustering.

6. Discussion and conclusions

In this paper, a GAGR has been developed for clustering. In the GAGR clustering algorithm, each chromosome represents the centers of the clusters by a sequence of real-valued numbers. This is more natural than the binary representation. In order to reduce the degeneracy caused by different chromosomes describing the same cluster result, a gene rearrangement of the chromosome has been defined. In addition, a new path-based crossover operator which builds a path from one chromosome to another chromosome has been presented. These two processes allow our GAGR clustering to explore the search space more effectively. Then, adaptive probabilities of crossover and mutation are used to prevent the GAGR clustering algorithm from getting stuck at a local optimal solution. The superiority of the GAGR clustering algorithm over KGA-clustering, GA-clustering and K-means algorithm has been demonstrated by the experiments. Moreover, the GAGR clustering has been applied to the multispectral remote sensing image for clustering the pixels into several classes, which also illustrated its effectiveness and superiority.

Our future works include:

1. The number of clusters cannot be determined by the GAGR clustering algorithm and this will limit its application in the real world. Future work should be done to develop a mean that allows an automatic identification of the number of clusters.
2. On the more theoretical side, an investigation of the path-based crossover operator is needed.

Acknowledgments

The authors would like to thank the Editor and the anonymous referees for their helpful comments and suggestions to improve the quality of the paper. The work was supported by the Cultivation Fund of the Key Scientific and Technical Innovation Project, Ministry of Education of China (No. 706004).

References

[1] B. Everitt, S. Landau, M. Leese, Cluster Analysis, Arnold, London, 2001.
[2] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Networks 16 (3) (2005) 645–678.
[3] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, Reading, MA, 1974.
[4] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999) 264–322.
[5] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley, New York, 2001.
[6] J. Handl, J. Knowles, D.B. Kell, Computational cluster validation in post-genomic data analysis, Bioinformatics 21 (15) (2005) 3201–3212.
[7] M. Ankerst, M. Breunig, H.P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, in: Proceedings of the International Conference on Management Data, 1999, pp. 49–60.
[8] M. Ester, H.P. Kriegel, J. Sander, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, 1996, pp. 226–231.
[9] S.Y. Lu, K.S. Fu, A sentence-to-sentence clustering procedure for pattern analysis, IEEE Trans. Syst. Man Cybern. 8 (1978) 381–389.
[10] E. Vorhees, The effectiveness and efficiency of agglomerative hierarchical clustering in document retrieval, Ph.D. Dissertation, Department of Computer Science, Cornell University, Ithaca, NY, 1985.
[11] T. Kohonen, Self-organization and Associative Memory, third ed., Springer, New York, Berlin, 1989.
[12] P.N. Suganthan, Hierarchical overlapped SOM's for pattern classification, IEEE Trans. Neural Networks 10 (1) (1999) 193–196.
[13] D. Brugger, M. Bogdan, W. Rosenstiel, Automatic cluster detection in Kohonen's SOM, IEEE Trans. Neural Networks 19 (3) (2008) 442–459.
[14] E.C.K. Tsao, J.C. Bezdek, N.R. Pal, Fuzzy Kohonen clustering networks, Pattern Recognition 27 (1994) 757–764.
[15] A.K. Qin, P.N. Suganthan, Robust growing neural gas algorithm with application in cluster analysis, Neural Networks 17 (8–9) (2004) 1135–1148.
[16] G.J. Mclachlan, K.E. Basford, Mixture Models: Inference and Applications to Clustering, Marcel Dekker, New York, 1988.
[17] G.J. Mclachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley, New York, 1997.
[18] S. Mitra, H. Banka, Multi-objective evolutionary biclustering of gene expression data, Pattern Recognition 39 (2006) 2464–2477.
[19] S.C. Madeira, A.L. Oliveira, Biclustering algorithms for biological data analysis: a survey, IEEE Trans. Comput. Biol. Bioinf. 1 (1) (2004) 24–45.
[20] Y. Kluger, R. Barsi, J.T. Cheng, M. Gerstein, Spectral biclustering of microarray data: coclustering genes and conditions, Genome Res. 13 (4) (2003) 703–716.
[21] Y. Hong, S. Kwong, Y.C. Chang, Q.S. Ren, Unsupervised feature selection using clustering ensembles and population based incremental learning algorithm, Pattern Recognition 41 (2008) 2742–2756.
[22] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2002) 583–617.
[23] A. Topchy, A.K. Jain, W. Punch, Clustering ensembles: models of consensus and weak partitions, IEEE Trans. Pattern Anal. Mach. Intell. 27 (12) (2005) 1866–1881.
[24] A. Ferligoj, V. Batagelj, Direct multicriterion clustering, J. Classification 9 (1992) 43–61.
[25] M.B. Dale, P.T. Dale, Classification with multiple dissimilarity matrices, Coenoses 9 (1992) 1–13.
[26] S.Z. Selim, M.A. Ismail, K-means type algorithms: a generalized convergence theorem and characterization of local optimality, IEEE Trans. Pattern Anal. Mach. Intell. 6 (1984) 81–87.
[27] H. Spath, Cluster Analysis Algorithms, Ellis Horwood, Chichester, UK, 1989.
[28] T. Kanungo, D. Mount, N.S. Netanyahu, C. Piatko, R. Silverman, A. Wu, An efficient K-means clustering algorithm: analysis and implementation, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 881–892.
[29] A. Likas, N. Vlassis, J.J. Verbeek, The global K-means clustering algorithm, Pattern Recognition 36 (2003) 452–461.
[30] D. Charalampidis, A modified K-means algorithm for circular invariant clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (12) (2005) 1856–1865.
[31] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[32] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs AI Series, Springer, New York, 1994.
[33] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Doctoral Dissertation, University of Michigan, Ann Arbor, MI, 1975.
[34] C.A. Murthy, N. Chowdhury, In search of optimal clusters using genetic algorithms, Pattern Recognition Lett. 17 (1996) 825–832.
[35] E. Falkenauer, Genetic Algorithms and Grouping Problems, Wiley, New York, NY, 1999.
[36] S. Bandyopdhyay, U. Maulik, An evolutionary technique based on K-means algorithm for optimal clustering in RN, Inf. Sci. 146 (2002) 221–237.
[37] S. Bandyopdhyay, S. Saha, GAPS: a clustering method using a new point symmetry-based distance measure, Pattern Recognition 40 (2007) 3430–3451.
[38] K. Krishna, M.N. Murty, Genetic K-means algorithm, IEEE Trans. Syst. Man Cybern. Part B Cybern. 29 (1999) 433–439.
[39] U. Maulik, S. Bandyopadhyay, Genetic algorithm based clustering technique, Pattern Recognition 33 (2000) 1455–1465.
[40] P. Scheunders, A genetic c-means clustering algorithm applied to color image quantization, Pattern Recognition 30 (6) (1997) 859–866.
[41] L.O. Hall, I.B. Özyurt, J.C. Bezdek, Clustering with a genetically optimized approach, IEEE Trans. Evol. Comput. 3 (2) (1999) 103–112.
[42] L.Y. Tseng, S.B. Yang, A genetic approach to the automatic clustering problem, Pattern Recognition 34 (2) (2001) 415–424.
[43] M. Laszlo, S. Mukherjee, A genetic algorithm using hyper-quadtrees for low-dimensional K-means clustering, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 533–543.

[44] M. Laszlo, S. Mukherjee, A genetic algorithm that exchanges neighboring centers for K-means clustering, Pattern Recognition Lett. 28 (2007) 2359–2366.
[45] G. Fleurya, A. Hero, S. Zareparsi, A. Swaroop, Gene discovery using Pareto depth sampling distributions, J. Franklin Inst. 341 (1–2) (2004) 55–75 (special number on Genomics, Signal Processing and Statistics).
[46] M. Morita, R. Sabourin, F. Bortolozzi, C.Y. Suen, Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition, in: Proceedings of the 7th International Conference on Document Analysis Recognition, 2003, pp. 666–671.
[47] P. Gancarski, A. Blanscheé, A. Wania, Comparison between two coevolutionary feature weighting algorithms in clustering, Pattern Recognition 41 (2008) 983–994.
[48] N. Radcliffe, P. Surry, Fitness Variance of Formae and Performance Prediction, Foundations of Genetic Algorithm 3, Morgan Kaufmann, San Mateo, 1995, pp. 51–72.
[49] E. Falkenauer, A hybrid grouping genetic algorithm for bin packing, J. Heuristics 2 (1996) 5–30.
[50] M. Srinivas, L. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms, IEEE Trans. Syst. Man Cybern. 24 (4) (1994) 656–667.
[51] ⟨http://www.ics.uci.edu/∼mlearn/MLRepository.html⟩.
[52] G.W. Milligan, M.C. Cooper, A study of standardization of variables in cluster analysis, J. Classification 5 (1988) 181–204.
[53] A. Hubert, Comparing partitions, J. Classification 2 (1985) 193–198.
[54] H. Frigui, R. Krishnapuram, A robust competitive clustering algorithm with applications in computer vision, IEEE Trans. Pattern Anal. Mach. Intell. 21 (1999) 450–465.
[55] M.S. Yang, K.L. Wu, A similarity-based robust clustering method, IEEE Trans. Pattern Anal. Mach. Intell. 26 (2004) 434–448.
[56] J.E. Jackson, A User's Guide to Principle Components, Wiley, New York, 1991.
[57] I.T. Jolliffe, Principal Components Analysis, Springer, New York, 1986.
[58] T.W. Anderson, An Introduction to Multivariate Statistical Analysis, Wiley, New York, 1984.
[59] S. Bandyopadhuay, U. Maulik, Multiobjective genetic clustering for pixel classification in remote sensing imagery, IEEE Trans. Geosci. Remote Sensing 45 (5) (2007) 1506–1511.

About the Author—DONGXIA CHANG received the B.S. and M.S. degrees in mathematics from Xidian University, in 2000 and 2003, respectively. She is currently pursuing the Ph.D. degree at the Department of Automation, Tsinghua University. Her current research interests include evolutionary computation, clustering and intelligent signal processing.

About the Author—XIANDA ZHANG received the B.S. degree in radar engineering from Xidian University, in 1969, the M.S. degree in instrument engineering from Harbin Institute of Technology in 1982, and the Ph.D. degree in electrical engineering from Tohoku University, Sendai, Japan, in 1987. Since 1992, he has been with the Department of Automation, Tsinghua University. His current research interests are signal processing with applications in radar and communications and intelligent signal processing.

About the Author—CHANGWEN ZHENG received the B.S. degree in mathematics and Ph.D. degree in control science and engineering from Huazhong Normal University, in 1992 and 2003, respectively. He was with the General Software Laboratory, Institute of Software, Chinese Academy of Sciences since 2003. His current research interests include route planning, evolutionary computation and neural networks.