PERGAMON

Pattern Recognition 33 (2000) 1251–1259

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

A genetic clustering algorithm for data with non-spherical-shape clusters[☆]

Lin Yu Tseng*, Shiueng Bien Yang

Department of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan 402, People's Republic of China

Received 13 October 1998; received in revised form 10 March 1999; accepted 17 March 1999

Abstract

In solving clustering problem, traditional methods, for example, the K-means algorithm and its variants, usually ask the user to provide the number of clusters. Unfortunately, the number of clusters in general is unknown to the user. The traditional neighborhood clustering algorithm usually needs the user to provide a distance d for the clustering. This d is difficult to decide because some clusters may be compact but others may be loose. In this paper, we propose a genetic clustering algorithm for clustering the data whose clusters are not of spherical shape. It can automatically cluster the data according to the similarities and automatically find the proper number of clusters. The experimental results are given to illustrate the effectiveness of the genetic algorithm. © 2000 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Clustering; Genetic clustering algorithm; Non-spherical-shape clusters

1. Introduction

The clustering problem is defined as the problem of classifying a collection of objects into a set of natural clusters without any a prior knowledge. For years, many clustering methods were proposed by many researchers, for example, see Refs. [1–6]. These methods can be basically classified into two categories: hierarchical and non-hierarchical. The hierarchical methods can be further divided into the agglomerative methods and the divisive methods. The agglomerative methods merge together the most similar clusters at each level and the merged clusters will remain in the same cluster at all higher levels. In the divisive methods, initially the set of all objects is viewed as a cluster and at each level, some clusters are binary divided into smaller clusters. There are also many non-hierarchical methods. Among them, the K-means algorithm is an important one. It is an iterative hill-climbing algorithm and the solution obtained depends on the initial clustering. Although the K-means algorithm had been applied to many practical clustering problems successfully, it is shown in Ref. [7] that the algorithm may fail to converge to a local minimum under certain conditions. In Ref. [8], a branch and bound algorithm was proposed to find the globally optimum clustering. However, it might take much computation time. In Refs. [9,10], simulated annealing algorithms for the clustering problem were proposed. These algorithms may find a globally optimum solution under some conditions. Most of these clustering algorithms require the user to provide the number of clusters as an input. But the user in general has no idea about the number of clusters. Hence, the user is forced to try different numbers of clusters when using these clustering algorithms. This is tedious and the clustering result may be no good especially when the number of clusters is large and not easy to guess. The K-means algorithm is not suitable for clustering the data whose clusters are not of spherical shape. In Ref. [11], a neighborhood clustering algorithm based on the mean distance from an object to its nearest

[☆]This research work was partially supported by the National Science Council of ROC under the contract NSC 87-2213-E-005-002.

* Corresponding author. Tel.: + 886-4-2874020; fax: + 886-4-2873028.

E-mail address: lytseng@amath.nchu.edu.tw (L.Y. Tseng).

neighbor was proposed. It can cluster this kind of data. But just like other neighborhood clustering methods, the threshold of distance for grouping objects together is difficult to decide. Some papers, for example, Refs. [12,13] had been devoted to this topic.

In this paper, we propose a genetic clustering algorithm for the data whose clusters may not be of spherical shape. Since genetic algorithm is good at searching, the clustering algorithm will automatically find the proper cluster number and classify the objects into these clusters at the same time.

The remaining part of the paper is organized as follows. In Section 2, the basic concept of the genetic approach is introduced. In Section 3, the genetic clustering algorithm for the data whose clusters may not be of spherical shape is described. The experimental results are given in Section 4 and we conclude the paper in Section 5.

2. The basic concept of the genetic strategy

The genetic strategy consists of an initialization step and the iterative generations. In each generation, there are three phases, namely, the reproduction phase, the crossover phase and the mutation phase. In the initialization step, a set of strings will be randomly generated. This set of strings is called the population. Each string consists of 0's and 1's. The meanings of the strings in the algorithm will be described in Section 3. After the initialization step, there is an iteration of generations. The user may specify the number of generations that he/she wants the genetic algorithm to run. After each generation, a set of strings with better fitness will be obtained and a clustering will thus be derived. The genetic algorithm will run the specified number of generations and retain the best clustering. The three phases in each generation are introduced in the following.

In the reproduction phase, the fitness of each string is calculated. The calculation of the fitness is the most important part of our algorithm. After the calculation of the fitness for each string in the population, the reproduction operator is implemented by using a roulette wheel with slots sized according to fitness. In the crossover phase, strings are chosen by pairs. For each chosen pair, two random numbers are generated to decide which pieces of the strings are to be interchanged. Suppose the length of the string is n, each random number is an integer in [1, n]. For example, if two random numbers are 2 and 5, position 2 to position 5 of this pair of strings are interchanged. For each chosen pair of strings, the crossover operator is applied by probability p_c. In the mutation phase, bits of the strings in the population will be chosen with probability p_m. Each chosen bit will be changed from 0 to 1 or from 1 to 0. In our experiments, we choose p_c and p_m to be 0.8 and 0.1 respectively.

3. The genetic clustering algorithm for data with non-spherical-shape clusters

There are data whose clusters are not of spherical shape. Some examples are given in Figs. 1–3. In this section, we describe a genetic clustering algorithm CLUSTERING for these kinds of data. Let there be n objects, $O_1, O_2, \dots, O_n$. The algorithm CLUSTERING consists of two stages. The first stage consists of the following steps.

Step 1: For each object O_i, find the distance between O_i and its nearest neighbor. That is,

$$d_{NN}(O_i) = \min_{j \neq i} \|O_j - O_i\|,$$

$$\text{where } \|O_j - O_i\| = \left(\sum_{q=1}^{p} (O_{jq} - O_{iq})^2 \right)^{1/2}.$$

Step 2: Compute d_{av}, the average of the nearest neighbor distances, as follows.

$$d_{av} = \frac{1}{n} \sum_{i=1}^{n} d_{NN}(O_i). \text{ Let } d = u^* d_{av}.$$

(d is decided by the parameter u and u is empirically chosen to be 1.5.)

Step 3: View the n objects as nodes of a graph. Compute the adjacency matrix $A_{n \times n}$ as follows.

$$A(i,j) = \begin{cases} 1 & \text{if } \|O_i - O_j\| d, \\ 0 & \text{otherwise,} \end{cases}$$

where $1 \leqslant j \leqslant i \leqslant n$.

Step 4: Find the connected components of this graph. Let these connected components be denoted by C_1, $C_2, \dots, C_m$.

The connected components $C_1, C_2, \dots, C_m$ obtained in the first stage will be taken as the initial clusters in the second stage. Basically, the second stage is a genetic algorithm, which will merge some of these C_i's if they are close enough to one another. We define the distance matrix $D_{m \times m}$ to specify the distance between each pair of clusters C_i and C_j.

$$D(i,j) = \min_{O_r \in C_i,\ O_s \in C_j} \|O_r - O_s\|.$$

The initialization step and the three phases of each generation of this genetic algorithm are described in the following.

Initialization step: A population of N strings is randomly generated. In our experiments, N is equal to 50. The length of each string is m, which is the number of the initial clusters obtained in the first stage. Each string represents a subset of $\{C_1, C_2, \dots, C_m\}$. If C_i is in this subset, the ith position of the string is 1; otherwise, it is 0. For example, suppose string R_1 represents the subset

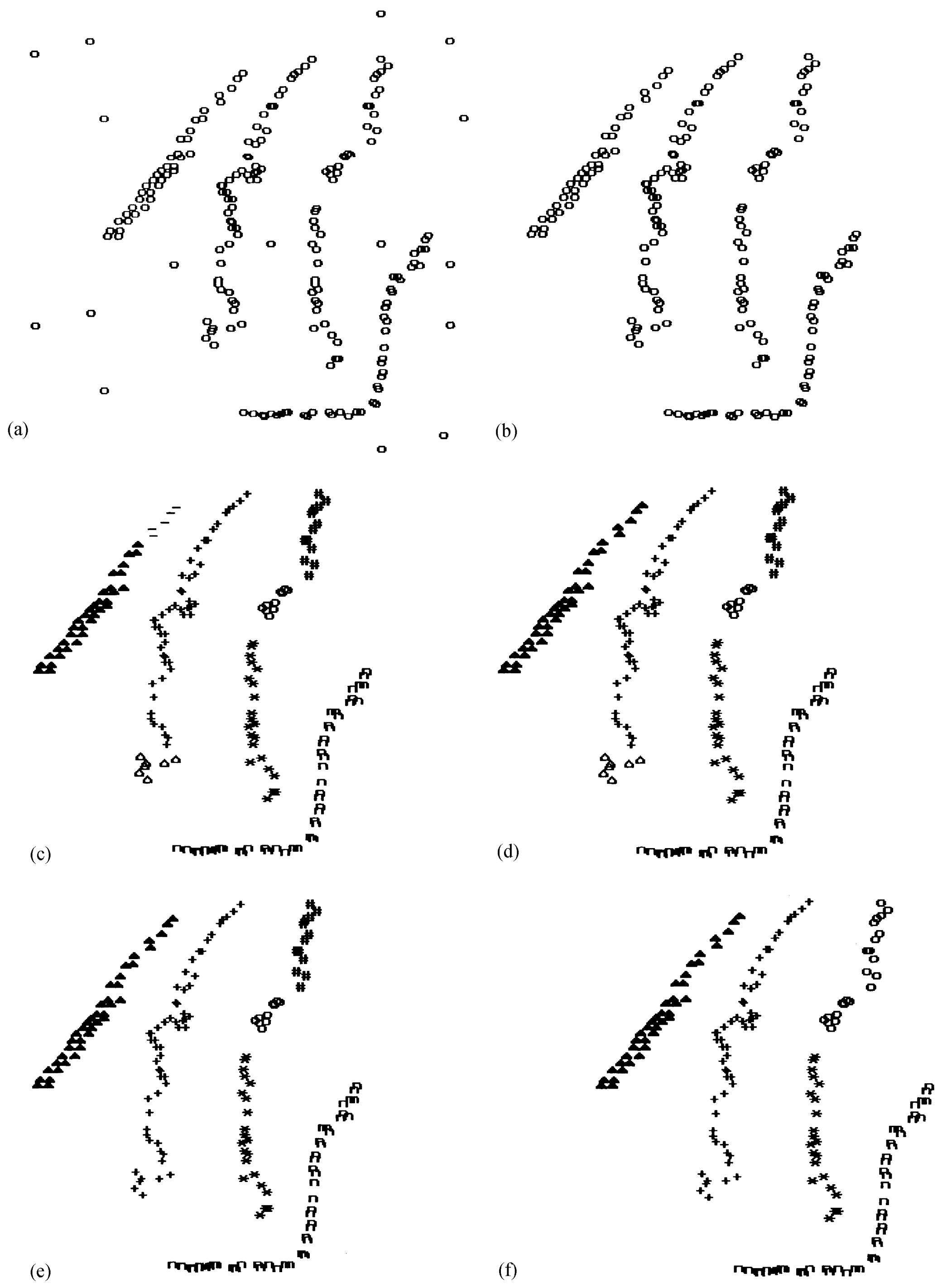

Fig. 1. The first set of test data and the clustering results. (a) The original test data. (b) The test data with noises removed. (c) 8 clusters. (d) 7 clusters. (e) 6 clusters. (f) 5 clusters. (g) 4 clusters. (h) 3 clusters. (i) 2 clusters. (j) 4 clusters obtained by applying the genetic algorithm to the test data. (k) 4 clusters obtained by applying K-means algorithm.

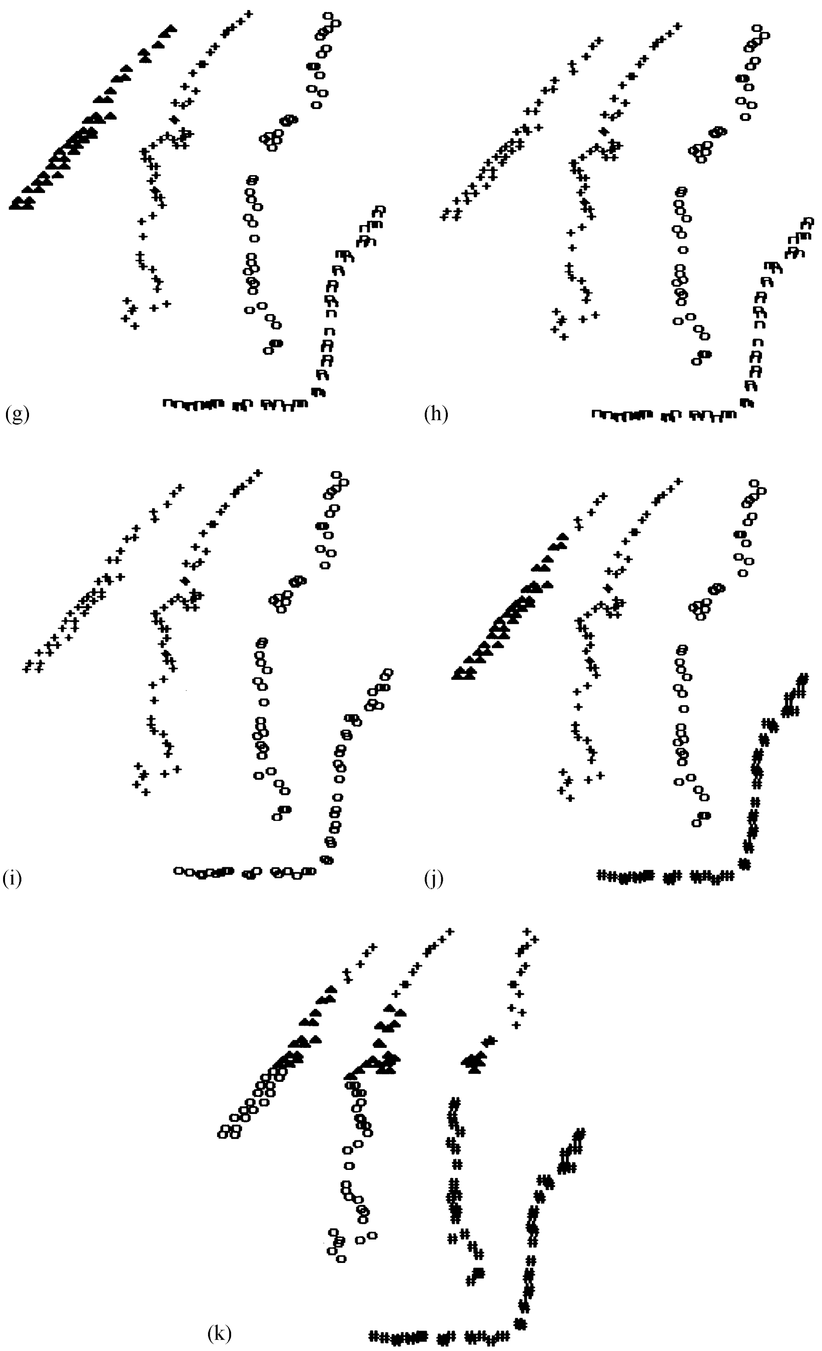

Fig. 1. (*Continued.*)

$\{C_1, C_2, C_3\}$ and string R_2 represents the subset $\{C_1, C_3, C_4\}$, then R_1 is $11100\ldots0$ and R_2 is $10110\ldots0$.

For each string R_i in the population, two sets U_i and U'_i are defined as follows:

$$U_i = \{j \mid \text{The } j\text{th bit of } R_i \text{ is } 1\},$$

$$U'_i = \{j \mid \text{The } j\text{th bit of } R_i \text{ is } 0\}.$$

That is, U_i contains those indices at which R_i has bit 1 and U'_i contains those indices at which R_i has bit 0. These two sets are used to define the intra-distance D_{intra} and the inter-distance D_{inter} in the following. Each string R_i represents a subset of $\{C_1, C_2, \ldots, C_m\}$. We define D_{intra} to represent the intra-distance among the clusters in this subset. We also define D_{inter} to represent

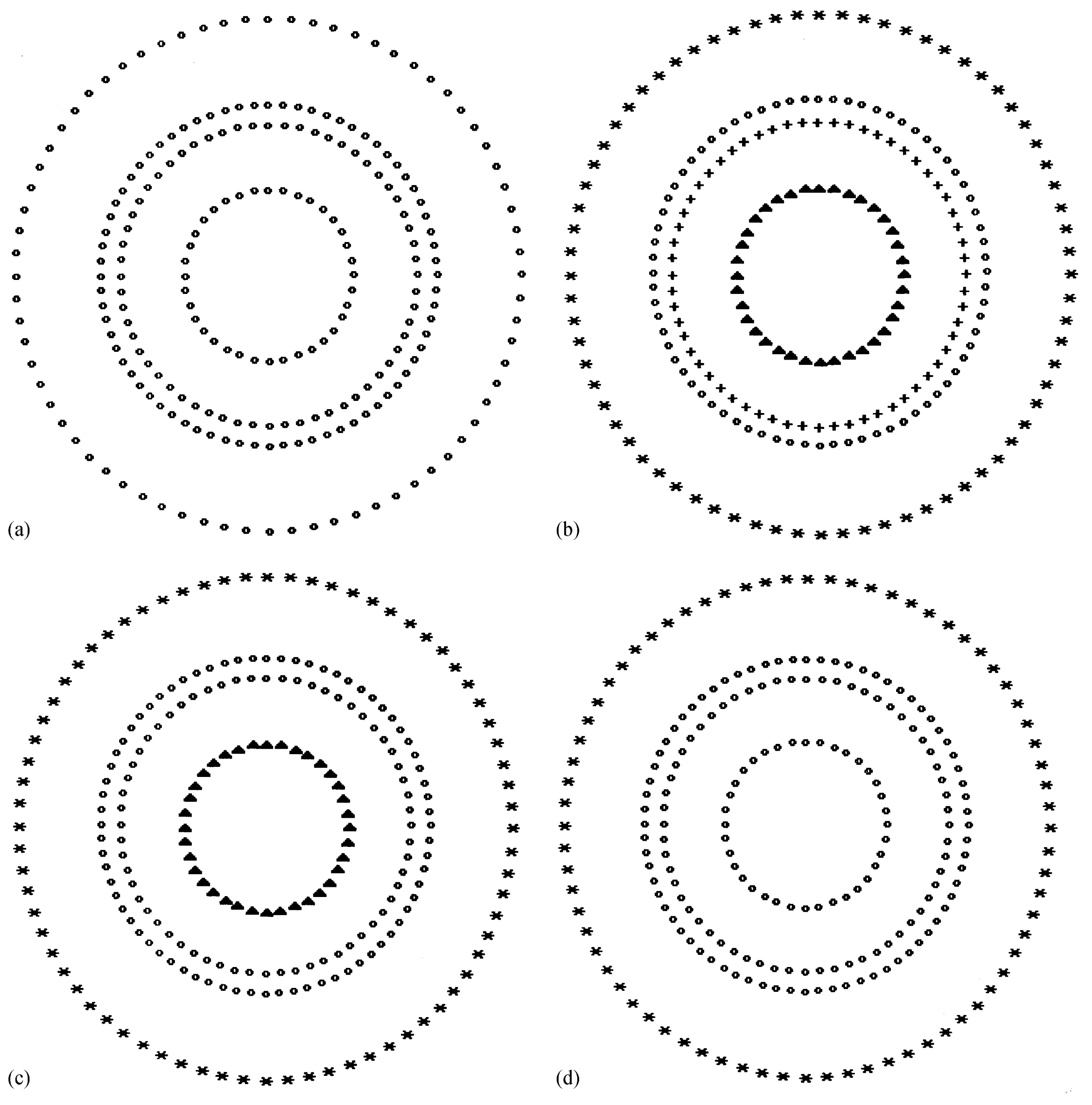

Fig. 2. The second set of test data and the clustering results. (a) The original test data. (b) 4 clusters. (c) 3 clusters. (d) 2 clusters.

the inter-distance between this subset and the set of all other clusters not in this subset.

$$D_{\text{intra}}(R_i) = \max_{j \in U_i} \min_{\substack{k \in U_i \\ k \neq j}} D(j, k),$$

$$D_{\text{inter}}(R_i) = \min_{\substack{j \in U_i \\ k \in U'_i}} D(j, k),$$

If R_i contains only 0's, both $D_{\text{intra}}(R_i)$ and $D_{\text{inter}}(R_i)$ are defined to be 0. If R_i contains only one 1, both $D_{\text{intra}}(R_i)$ and $D_{\text{inter}}(R_i)$ are defined to be 0. Some explanations may be helpful in understanding the definitions of $D_{\text{intra}}(R_i)$ and $D_{\text{inter}}(R_i)$. Suppose R_i represents $\{C_1, C_2, C_3\}$ which is a subset of $\{C_1, C_2, \ldots, C_m\}$, for each C_j, there must be a C_k that is nearest to C_j in the subset. Suppose Fig. 4(a) indicates these three clusters C_1, C_2 and C_3, the nearest clusters to C_1, C_2 and C_3 are C_2, C_1 and C_2 respectively. For each pair of nearest clusters, there is a distance and D_{intra} is just the maximum of all these distances. In Fig. 4(a), $D_{\text{intra}}(R_i)$ is $D(2, 3)$. Therefore, $D_{\text{intra}}(R_i)$ is used to measure the nearness of the clusters in the subset represented by R_i. As indicated in Fig. 4(b), suppose the clusters that are outside this subset and nearest to C_1, C_2 and C_3 are C_7, C_4 and C_5 respectively. Then, $D_{\text{inter}}(R_i)$ is

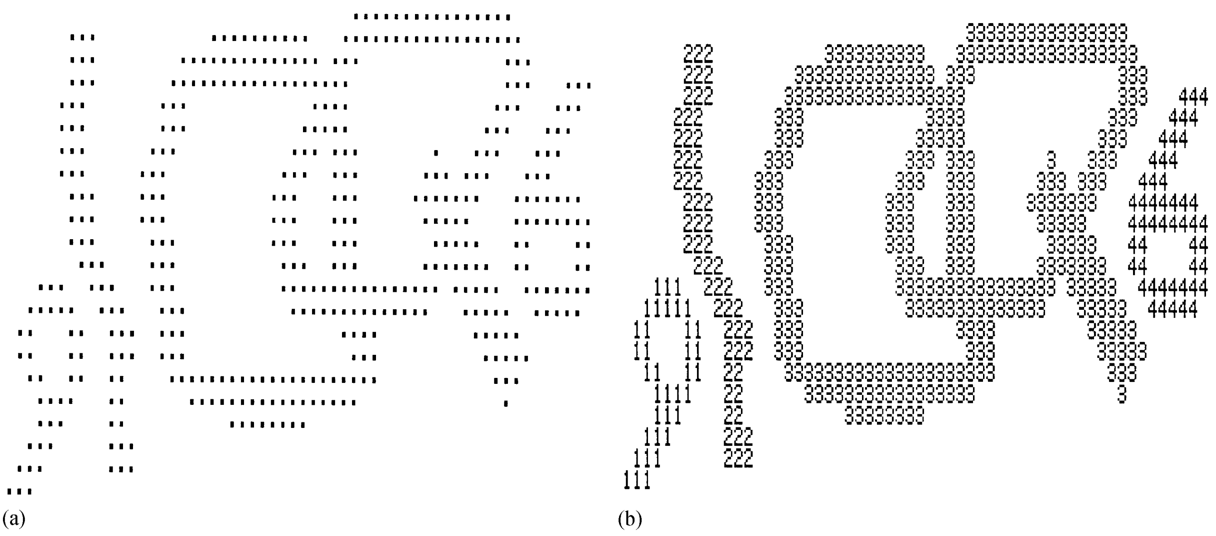

Fig. 3. The third set of test data and the clustering result. (a) The third set of test data. (b) The clustering result.

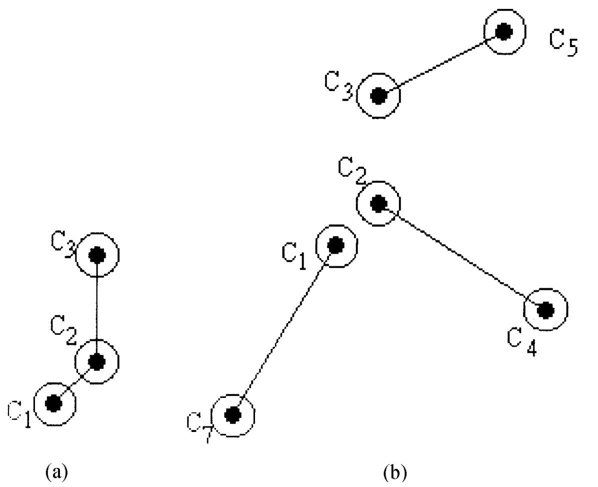

Fig. 4. An example to illustrate the definitions of D_{intra} and D_{inter}.

defined to be the smallest one of these three distances, that is, $D(3, 5)$. So $D_{\text{inter}}(R_i)$ is used to measure the degree of separation among the subset represented by R_i and other clusters.

Reproduction phase: The fitness of the string R_i is defined as follows:

$$\text{SCORE}(R_i) = D_{\text{inter}}(R_i)^* w - D_{\text{intra}}(R_i),$$

where w is a weight. If the value of w is small, we emphasize the importance of $D_{\text{intra}}(R_i)$. That is, only clusters very near to one another are merged. This tends to produce more clusters and each cluster tends to be compact. If the value of w is chosen to be large, we emphasize the importance of $D_{\text{inter}}(R_i)$. That is, clusters not very near to one another may be merged in order to make the distances among the merged ones larger. This tends to produce lesser clusters and each cluster tends to be loose. According to our experience, the value of w is within [1, 3]. After the calculation of fitness for each string in the population, the reproduction operator is implemented by using a roulette wheel with slots sized according to fitness. That is, R_i is reproduced by the probability $\text{SCORE}(R_i)/\sum_{i=1}^{N}\text{SCORE}(R_i)$.

Crossover phase: In the crossover phase, for each chosen pair of strings, two random numbers in $[1, m]$ are generated to decide which pieces of the strings are to be interchanged. For example, suppose $R_i = 1010010\ldots0$, $R_j = 1100110\ldots0$, and two random numbers chosen are 3 and 6, then position 3 to position 6 of two strings are interchanged and final R_i and R_j are $1000110\ldots0$ and $1110010\ldots0$, respectively. For each chosen pair of strings, the crossover operator is done with probability p_c, which is equal to 0.8 in our experiments.

Mutation phase: In the mutation phase, bits of the strings in the population will be chosen with probability p_m, which is equal to 0.1 in our experiments. Each chosen bit will be changed from 0 to 1 or from 1 to 0. For example, suppose $R_i = 1010010\ldots0$ and the second bit of R_i is chosen to do the mutation, then $R_i = 1110010\ldots0$ after the mutation phase.

In each generation of this genetic algorithm, what we really want is not the string with the best fitness but a set of strings with better fitness. This set represents a good merging method for $C_1, C_2, \ldots, C_m$. By applying this good merging method to merge some of

C_i's, a good clustering will be obtained. The Algorithm Merge_Sets_Finding consists of the following four steps is used to find the sets of C_i's that are to be merged. This algorithm is executed after the calculation of fitness for each string and before the application of the reproduction operator in the reproduction phase.

Step 1: Sort the fitness of the strings in non-increasing order. For the sake of brevity, let us assume $\mathrm{SCORE}(R_1) \geqslant \mathrm{SCORE}(R_2 \geqslant \cdots \geqslant \mathrm{SCORE}(R_N)$.

$i = 1$: $U = \phi$.

Step 2: Choose R_i. Let $V_i = \{C_j | j \in U_i\}$. $U_i = \{j|\text{The } j\text{th bit of } R_i \text{ is } 1\}$ as defined earlier in this section. The clusters in V_i are to be merged.

$U = U \cup U_i$.

Step 3: Choose smallest $l > i$ such that $U_l \cap U = \phi$. If no such l exists then go to Step 4 else $i = l$ and go to Step 2.

Step 4: End.

An example to illustrate the Merge_Sets_Finding algorithm is given as follows. Suppose $\mathrm{SCORE}(R_1) \geqslant \mathrm{SCORE}(R_2) \geqslant \cdots \geqslant \mathrm{SCORE}(R_N)$, R_1 represents subset $\{C_1, C_2, C_3\}$, R_2 represents subset $\{C_3, C_4, C_6\}$, R_3 represents subset $\{C_4, C_5\}$, and each of R_4 to R_N represents a subset containing at least one of C_1, C_2, C_3, C_4 and C_5, then in this algorithm, by first choosing R_1, clusters C_1, C_2, C_3 are merged. Since the subset represented by R_2 contains C_3 which is already in the subset represented by R_1, hence R_2 is discarded. After that, R_3 is considered, the subset represented by R_3 contains no clusters that are already merged. Hence the clusters C_4, C_5 in this subset are merged. Since each of R_4 to R_N represents a subset containing at least one cluster that is already merged, they are all discarded.

The time complexity is analyzed as follows. In the first stage, Step 1 takes $O(n^2)$ time to calculate the distances between pairs of objects and takes $O(n)$ time to find the minimum. Step 2 takes $O(n)$ time to calculate the minimum. Step 3 takes $O(n^2)$ time to derive the adjacency matrix and Step 4 also takes $O(n^2)$ time to find the connected components by scanning the adjacency matrix. Therefore, the first stage spends $O(n^2)$ time. Before the second stage, we need to calculate $D(i,j)$ for all C_i, C_j, this takes $O(n^2)$ time in the worst case. The time complexity of the second stage is dominated by the calculation of $D_{\text{intra}}(R_i)$ and $D_{\text{inter}}(R_i)$. It takes $O(Nm^2)$ time in the worst case. Suppose the genetic algorithm is asked to run k generations, then the time complexity will be $O(kNm^2)$. Hence, the time complexity of the whole clustering algorithm is $O(n^2 + kNm^2)$. In our experiments, k equals 20 and N equals 50. In general, m is also a small number. So these three numbers can almost be taken as constants and the time complexity is then $O(n^2)$.

4. Experiments

Three sets of data are used to test the effectiveness of the clustering algorithm. Fig. 1(a) shows the original test data. There are some noises in the test data. By calculating the average of the nearest neighbor distances d_{av}, if an object has a distance $2d_{av}$ from its nearest neighbor, this object is taken as a noise and is discarded. Fig. 1(b) shows the test data with noises removed. As mentioned before, u is chosen to be 1.5 by experience. In our experiments on the first and the second sets of data, two other values of u, namely 1.2 and 2, are also used to illustrate that with a suitable choice of the value of w, a good clustering can be found with all three values of u. This means that u may be chosen from an interval, e.g. [1.2, 2], and the exact value of u is not crucial to the clustering result. In Table 1, if u is 1.2, there are eight initial clusters, which are shown in Fig. 1(c). If u is 1.5 or 2, there are five initial clusters, which are shown in Fig. 1(f). As shown in Table 1, several values of w are chosen from [1, 3]. For each value of w, the number of clusters and the maximum intra-distance are recorded. For example, if u is 1.2 and w is 2, there are four clusters as shown in Fig. 1(g) and the maximum intra-distance is 12.9. The values of w are chosen from [1, 3] by binary search. The numbers in circles in Table 1 indicate the sequence of chosen values of w. A good clustering can be found by conducting this binary search. The criteria of selecting the good clustering are as follows. Find the smallest w such that $N(w) = N(w') + 1$ and $D(w')/D(w) \geqslant 2$, where w' is next to w and larger than w. Then the clustering obtained by using this w is the good clustering. If there is no such case,

Table 1
Numbers of clusters and the maximum intra-distances of the first test set for different values of u and w

u	# components	$(N(w), D(w))$						
		$w = 1$	$w = 1.25$	$w = 1.5$	$w = 1.75$	$w = 2$	$w = 2.5$	$w = 3$
1.2	8	① (8, 6.5)	—	④ (6, 8.4)	⑥ (5, 8.8)	③ (4, 12.9)	⑤ (3, 29.1)	② (2, 31.2)
1.5	5	① (5, 8.8)	—	④ (4, 12.9)	⑤ (3, 29.1)	③ (2, 31.2)	—	② (2, 31.2)
2	5	① (5, 8.8)	—	④ (4, 12.9)	⑤ (3, 29.1)	③ (2, 31.2)	—	② (2, 31.2)

Table 2
Numbers of clusters and the maximum intra-distances of the second test set for different values of u and w

u	# components	$(N(w), D(w))$ $w = 1$	$w = 1.5$	$w = 2$	$w = 2.5$	$w = 3$
1.2	76	① (76, 6.9)	④ (3, 9.5)	③ (2, 19.5)	—	② (2, 19.5)
1.5	4	① (4, 8.9)	④ (3, 9.5)	③ (2, 19.5)	—	② (2, 19.5)
2	3	① (3, 9.5)	—	—	—	② (2, 19.5)

we find all w in [1, 3] such that $N(w) = N(w') + 1$ and $D(w')/D(w) \geqslant 1.5$. All clustering corresponding to these w's are output. If there are still no such cases, the clustering obtained by choosing 3 as the value of w is output. Fig. 1(c)–(i) show clustering with different number of clusters. In the first test data, the good clustering has four clusters as shown in Fig. 1(g). If we apply only the nearest neighbor clustering method (with $u = 2$), we obtain a clustering with five clusters as shown in Fig. 1(f), which is not very proper. If we apply only the genetic algorithm (i.e. the second stage of our clustering algorithm) directly to the test data, four clusters are obtained as shown in Fig. 1(j). The clustering is also not good. By applying the K-means algorithm with number of clusters equals to four, the clustering is shown in Fig. 1(k). This clustering result is bad because the K-means algorithm is not suitable for non-spherical-shape clusters. Fig. 2 and Table 2 show the second test data and its clustering results. A good clustering is the case of three clusters shown in Fig. 2(c). Fig. 3 shows the third set of test data and its clustering result.

5. Concluding remarks

A genetic clustering algorithm CLUSTERING is proposed. CLUSTERING is a clustering algorithm for data whose clusters may not be of spherical shape. Unlike the K-means algorithm which needs the user to provide it with the number of clusters, CLUSTERING can automatically search for a proper number as the number of clusters. By binary searching some proper interval for the value of w, a proper number of clusters and a good clustering can be found. In general, a natural and steady clustering will correspond to a significantly long interval of w values. The traditional neighborhood clustering algorithm usually needs the user to provide a distance d for the clustering. But a unique d for a set of objects often causes problems because there may be some natural clusters in which the objects are not close to one another within the distance d. CLUSTERING avoids this kind of problem by processing the data in a global view.

6. Summary

The clustering problem is a very important problem and has attracted much attention of many researchers. Some traditional methods, for example, the K-means algorithm and its variants, usually ask the user to provide the number of clusters. Unfortunately, the number of clusters in general is unknown to the user. Hence, the user usually has to try several times in order to get a good clustering. The traditional neighborhood clustering algorithm usually needs the user to provide a distance d for the clustering. This d is difficult to decide because some clusters may be compact but others may be loose. In this paper, a genetic clustering algorithm CLUSTERING is proposed. CLUSTERING is a clustering algorithm for data whose clusters may not be of spherical shape. Unlike the K-means algorithm, CLUSTERING can automatically search for a proper number as the number of clusters. By binary searching some proper interval for the value of w, which is the weighting factor between the inter-distance and the intra-distance of the clusters, a proper number of clusters and a good clustering can be found. Several experiments are conducted to illustrate the effectiveness of the genetic clustering algorithm.

References

[1] M.R. Anderberg, Cluster Analysis for Applications, Academic Press, New York, 1973.
[2] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addision-Wesley, Reading, MA, 1974.
[3] J.A. Hartigan, Clustering Algorithms, Wiley, New York, 1975.
[4] K.S. Fu, Communication and Cybernetics: Digital Pattern Recognition, Springer, Berlin, 1980.
[5] R. Dubes, A.K. Jain, Clustering Methodology in Exploratory Data Analysis, Academic Press, New York, 1980.
[6] P.A. Devijver, J. Kittler, Pattern Recognition-A Statistical Approach, Prentice-Hall, London, 1982.
[7] S.Z. Selim, M.A. Ismail, K-means-type algorithm: generalized convergence theorem and characterization of local optimality, IEEE Trans. Pattern Anal. Mach. Intell. 6 (1) (1984) 81–87.

[8] W.I. Koontz, P.M. Narendra, K. Fukunaga, A branch and bound clustering algorithm, IEEE Trans. Comput. c- 24 (9) (1975) 908–915.

[9] S.Z. Selim, K.S. Al-Sultan, A simulated annealing algorithm for the clustering problem, Pattern Recognition 24 (10) (1991) 1003–1008.

[10] R.W. Klein, R.C. Dubes, Experiments in projection and clustering by simulated annealing, Pattern Recognition 22 (2) (1989) 213–220.

[11] P.-Y. Yin, L.-H. Chen, A new non-iterative approach for clustering, Pattern Recognition Letters 15 (2) (1994) 125–133.

[12] G.C. Osbourn, R.F. Martinez, Empirically defined regions of influence for clustering analyses, Pattern Recognition 28 (11) (1995) 1793–1806.

[13] P.S. Stephen, Threshold validity for mutual neighborhood clustering, IEEE Trans. Pattern Anal. March. Intell. 15 (1) (1993) 89–92.

About the Author—LIN YU TSENG received the B.S. degree in mathematics from the National Taiwan University, Taiwan, in 1975, and the M.S. degree in computer science from the National Chiao Tung University, Taiwan, 1978. After receiving the M.S. degree, he worked in industry and taught at the university for several years. He received the Ph.D degree in computer science from National Tsing Hua University, Taiwan, in 1988. He is presently a Professor at the Department of Applied Mathematics, National Chung Hsing University. His research interests include pattern recognition, document processing, speech coding and recognition, neural networks and algorithm design.

About the Author—SHIUENG BIEN YANG received the B.S. degree in 1993 from the Department of Applied Mathemaics, National Chung Hsing University. He is now working towards the Ph.D. degree in the same department. His research interests include pattern recognition, speech coding and recognition, image coding and neural networks.