A Weighted Sum Validity Function for Clustering With a Hybrid Niching Genetic Algorithm

Weiguo Sheng, Stephen Swift, Leishi Zhang, and Xiaohui Liu

***Abstract*—Clustering is inherently a difficult problem, both with respect to the construction of adequate objective functions as well as to the optimization of the objective functions. In this paper, we suggest an objective function called the Weighted Sum Validity Function (WSVF), which is a weighted sum of the several normalized cluster validity functions. Further, we propose a Hybrid Niching Genetic Algorithm (HNGA), which can be used for the optimization of the WSVF to automatically evolve the proper number of clusters as well as appropriate partitioning of the data set. Within the HNGA, a niching method is developed to preserve both the diversity of the population with respect to the number of clusters encoded in the individuals and the diversity of the subpopulation with the same number of clusters during the search. In addition, we hybridize the niching method with the k-means algorithm. In the experiments, we show the effectiveness of both the HNGA and the WSVF. In comparison with other related genetic clustering algorithms, the HNGA can consistently and efficiently converge to the best known optimum corresponding to the given data in concurrence with the convergence result. The WSVF is found generally able to improve the confidence of clustering solutions and achieve more accurate and robust results.**

***Index Terms*—Cluster validity, clustering, evolutionary computation, genetic algorithms, niching methods.**

I. Introduction

CLUSTER analysis is a tool that attempts to assess the relationships among objects of the data set by organizing the objects into groups or clusters such that objects within a cluster are similar to each other but dissimilar from objects in other clusters. Many clustering algorithms [3], [5], [12], [21], [23], [27] have been developed to accomplish this. Generally, clustering methods can be divided into two main categories [25]: hierarchical and partitional. In hierarchical clustering, the number of clusters need not be specified *a priori*, and problems due to initialization and local optima do not arise. However, hierarchical methods consider only local neighbors in each step. The global shape and size of clusters are always ignored. Moreover, they are static methods, and objects committed to a given cluster in the early stages cannot move to a different cluster.

The methods for considering the global shape and size of clusters are the well-known partitional clustering algorithms. Partitional clustering algorithms can be divided into two classes [31]: hard (or crisp) clustering, where each data point belongs to only one cluster, and overlapping clustering, where every data point belongs to every cluster with a certain degree. Partitional algorithms are dynamic, and objects can move from one cluster to another. However, most partitional approaches are based on objective functions, which assume that the number of clusters is known *a priori*. Moreover, they use the alternating optimization technique, whose iterative nature makes them sensitive to initialization and susceptible to local optima.

Since *a priori* knowledge is generally not available, estimation of the number of clusters from the observed data set is required. This problem is related to the cluster validity and is essentially not resolved. The classical approach of determining the number of clusters has been to apply a given clustering algorithm for a range of K values and to evaluate a certain validity function of the resulting partitioning in each case [3], [16], [29]. The partitioning exhibiting the optimal validity is chosen as the true partitioning. This method for searching an optimal cluster number depends on the selected clustering algorithm, whose performance may depend on the initial cluster centers. Furthermore, the validity measures may be unreliable because the performance of the validity function may be sensitive to structures of different problem instances. Different validity functions may result in rather different solutions on the same set of resulting partitionings and there is no clear indication which of the different solutions is the best. Some other techniques of estimating the number of clusters in a data set are based on the idea of cluster removal and/or merging. In progressive clustering [9], [29], [30], the number of clusters is overspecified. After convergence, spurious clusters are eliminated, compatible clusters are merged, and "good" clusters are identified. An alternative version of the progressive clustering technique is to seek one cluster at a time until no more "good" clusters can be found [26], [46], [48]. These techniques may be more efficient than the classical approach; however, they are also dependent on the validity functions, which are used to evaluate the individual clusters.

Since the global optimum of the validity functions would correspond to the most "valid" solutions with respect to the functions, stochastic clustering algorithms based on Genetic Algorithms (GAs) have been reported to optimize the validity functions to determine the cluster number and partitioning of a data set simultaneously [14], [34], [35]. Other than evaluating the static clusters generated by a specific clustering algorithm, the validity functions in this approach are used as clustering objective functions for computing fitness, which guides the evolution to search for the "valid" solution. As a result, this approach does not depend on the specific clustering algorithm. However, it is still dependent on the validity functions. Additionally, this approach is also dependent on the genetic

Manuscript received August 16, 2004. This work was supported in part by the BBSRC, U.K., under BIO14300. This paper was recommended by Editor Fernando Comide.

The authors are with the Department of Information Systems and Computing, Brunel University, Uxbridge, London, UB8 3PH, U.K. (e-mail: weiguo.sheng@brunel.ac.uk; stephen.swift@brunel.ac.uk; leishi.zhang@brunel.ac.uk; xiaohui.liu@brunel.ac.uk).

Digital Object Identifier 10.1109/TSMCB.2005.850173

clustering algorithms. The genetic clustering algorithms used in [14], [34], and [35] may have difficulty in identifying the best known optimum corresponding to the given data set because they employ either the Simple GA (SGA) [22] or its variants, which may suffer from premature convergence to local optima. Furthermore, they may take a large amount of time to converge.

Since there is no "perfect" validity function that is likely to provide consistently reliable results across structures of different problem instances, in this paper, we address this dilemma by suggesting a function called Weighted Sum Validity Function (WSVF), which is a weighted sum of the several normalized validity functions, to conduct some sort of voting among its component functions about the best solution. Using more than one validity function via a weighted sum approach tends to increase the confidence of clustering solutions. A related motivation of using the WSVF is to build a robust function that can perform well over a wide range of data sets. Further, we propose a Hybrid Niching Genetic Algorithm (HNGA), which can be used to optimize the WSVF to automatically evolve the proper number of clusters and appropriate partitioning of the data set. Within the HNGA, a niching method is developed to prevent premature convergence by preserving both the diversity of the population with respect to the number of clusters encoded in the individuals and the diversity of the subpopulation with the same number of clusters during the search. Additionally, in order to improve the computational efficiency, we hybridize the niching method with the computationally attractive k-means.

The organization of the paper is as follows. After reviewing some cluster validity functions related to our work, we suggest the Weighted Sum Validity Function in Section II. Then, in Section III, we present the Hybrid Niching Genetic Algorithm. Section IV describes the six data sets employed in this work, followed by the discussion of the parameter settings of the HNGA. In the experiments in Section V, we show the effectiveness of both the HNGA and the WSVF. We finalize the paper with summaries and results in Section VI.

II. Validity Functions and the WSVF

In this section, we describe several cluster validity functions that have been chosen as components of the Weighted Sum Validity Function (WSVF). This is followed by the explanation of the WSVF.

Cluster validity is concerned with checking the quality of clustering results. At the partitional level, cluster validity evaluates the groupings of data into clusters: does any grouping reflect the "actual structure" of the data? Which grouping is "better"? It has been mainly used to estimate the number of clusters presented in the data set. Less commonly, cluster validity has been used for other purposes. In [2], a validity function is used in synergy with the fuzzy c-means objective function to guide the clustering process toward improved partitioning. In [14], [34], and [35], the validity functions are used as clustering objective functions to determine the cluster number and partitioning of the data set simultaneously. In this paper, we also use the validity functions to generate partitioning as well. However, unlike [14], [34], and [35], which is based on a certain validity function whose performance may be sensitive to structures of different problem instances, we suggest the WSVF, which is a weighted sum of the several normalized validity functions.

A number of validity functions have been proposed in the literature. Two decades ago Hubert and Arabie said in a paper on this topic "We will not try to review this literature comprehensively since that task would require the length of a monograph" [24]. In general, methods developed to validate partitioning can be classified into two categories: fuzzy cluster validity and hard cluster validity to evaluate fuzzy partitioning and hard partitioning, respectively. We concentrate on hard cluster validity, among which five well-known validity functions, namely, Davies–Bouldin (DB) [10], Silhouette statistic (SIL) [27], Dunn's function [13], a version of Generalized Dunn's function [4], and Calinski Harabasz (CH) [6], as well as a recently developed PBM index (also called I index) [39], have been chosen as the components of the WSVF in this paper. These six functions have rather different properties and rationales and should serve as an adequate basis for our purpose. However, the choice of the component functions for WSVF is flexible, and its performance can be improved by using other more effective combinations of validity functions. Intuitively, the component functions of a good combination should have different properties and rationales and favor different data distributions.

A. Six Validity Functions

DB: This function is the ratio of the sum of within-cluster scatter to between-cluster separation. Since scatter matrices depend on the geometry of the clusters, this function has both a statistical and geometric rationales. The scatter within the ith cluster S_i is defined as $S_i = 1/|C_i| \sum_{x \in C_i} \|x - z_i\|^2$, and the distance between cluster C_i and C_j is defined as $d_{ij} = \|z_i - z_j\|^2$. Here, $|C_i|$ is the number of objects in cluster C_i, and z_i represents the ith cluster center. The distance measure between the centers of clusters i and j is based on the squared Euclidean metric (the squared Euclidean metric is used to measure the dissimilarity or distance in this work unless otherwise mentioned). Then, we define $R_i = \max_{j, j \neq i} \{(S_i + S_j)/d_{ij}\}$. Now, the DB is defined as

$$\text{DB} = \frac{1}{K} \sum_{i=1}^{K} R_i \tag{1}$$

where K is the number of clusters. Data partitioning that minimizes the DB function will be treated as proper clustering result.

SIL: Let a_i denote the average distance between i and all other observations in the cluster to which i belongs. For any other cluster C, let $d(i, C)$ denote the average distance of i to all objects of C and b_i denote the smallest of these $d(i, C)$. The silhouette width of object i is $\text{Sil}_i = (b_i - a_i)/\max(a_i, b_i)$, and the overall average silhouette width is simply the average of Sil_i over all objects i

$$\overline{\text{SIL}} = \sum_i \frac{\text{Sil}_i}{n}. \tag{2}$$

Intuitively, objects with large silhouette width Sil_i are well clustered, whereas those with small Sil_i tend to lie between clusters.

Kaufman and Rousseeuw [27] suggested estimating the partitioning by that which gives the largest average silhouette width $\overline{\text{SIL}}$.

V_D: Let S and T be two nonempty subsets of X. Then, the diameter Δ of S and set distance δ between S and T are $\Delta(S) = \underbrace{\max}_{x,y\in S}\{d(x,y)\}$ and $\delta(S,T) = \underbrace{\min}_{x\in S, y\in T}\{d(x,y)\}$, respectively, where $d(x,y)$ is the distance between objects x and y. For any partitioning, Dunn's function is defined as

$$V_\text{D} = \underbrace{\min}_{1\le i\le K}\left\{\underbrace{\min}_{1\le j\le K, j\ne i}\left\{\frac{\delta(C_i,C_j)}{\underbrace{\max}_{1\le k\le K}\{\Delta(C_k)\}}\right\}\right\}. \quad (3)$$

V_{33}: Bezdek *et al.* [4] presented a generalized Dunn's function, which may have 18 different forms depending on the functional forms used to select δ_i and Δ_j. Here, we use the combination of δ_3 and Δ_3, which is recommended as one of the most reliable forms. The δ_3 and Δ_3 are defined as $\Delta_3(S) = 2\left(\sum_{x\in S} d(x,v_S)/|S|\right)$ and $\delta_3(S,T) = (1/(|S||T|))\sum_{x\in S, y\in T} d(x,y)$, respectively, where $V_S = (1/|S|)\sum_{x\in S} x$. Then, the generalized Dunn's function V_{33} is defined as

$$V_{33} = \underbrace{\min}_{1\le i\le K}\left\{\underbrace{\min}_{1\le j\le K, j\ne i}\left\{\frac{\delta_3(C_i,C_j)}{\underbrace{\max}_{1\le k\le K}\{\Delta_3(C_k)\}}\right\}\right\}. \quad (4)$$

Large values of V_D and V_{33} correspond good partitioning.

CH: This function performed the best in Milligan and Cooper's [37] simulation comparison of 30 different procedures. For n data objects and K clusters, it is defined as

$$\text{CH} = \frac{\frac{\text{trace}B}{(K-1)}}{\frac{\text{trace}W}{(n-1)}} \quad (5)$$

where trace$B$ and trace$W$ are the between and within cluster sums of squares. They can be written as trace$B = \sum_{k=1}^{K} n_k \|z_k - z\|^2$ and trace$W = \sum_{k=1}^{K}\sum_{i=1}^{n_k}\|x_i - z_k\|^2$, where n_k is the number of objects in cluster C_i, and z is the center of entire data set. Large value of CH corresponds good partitioning.

PBM: The PBM index is defined as follows:

$$\text{PBM} = \left(\frac{1}{K}\times\frac{E_1}{E_k}\times D_k\right)^2 \quad (6)$$

where E_1 is constant for a given data set. Here, $E_k = \sum_{k=1}^{K}\sum_{i=1}^{n_k}\|x_i - z_k\|^2$, and $D_k = \max_{i,j=1}^{K}\|z_i - z_j\|^2$, where n_i is the number of objects in cluster C_i, and z_k is the center of the kth cluster. The best partitioning occurs at the maximum value of this function.

B. Weighted Sum Validity Function

The WSVF consists of adding all the above functions together using a weighting coefficient for each one. This means that the optimization problem of the above functions can be transformed into a combinational optimization problem of the form

$$\max \text{WSVF} = \sum_{i=1}^{m} w_i f_i(x) \quad (7)$$

where m is the number of component functions, specifically $m = 6$ used in this paper, w_i are the non-negative weighting coefficients representing the relative importance of the functions such that $\sum_{i=1}^{m} w_i = 1$, and $f_i(x)$ are component functions corresponding to the 1/DB, SIL, V_D, V_{33}, CH, and PBM, respectively. Note that maximization of the 1/DB will ensure minimization of the DB function. Since there is no *a priori* information about the relative importance of any individual function, we set the weighting coefficients as $w_1 = w_2 = \cdots = w_\text{m} = 1/m$.

Note that the values of different functions may have different order of magnitude. Using a weighted sum approach to calculate the WSVF, it may be dominated by the functions with large values. For example, if $f_1(x)$ represents the value with $1000 < f_1(x) < 2000$ and $f_2(x)$ represents the value with $0 < f_2(x) < 1$, then $w_1 f_1(x) + w_2 f_2(x)$ may be dominated by $f_1(x)$. Therefore, if we want w_i to closely reflect the importance of the functions, all functions should be expressed in units of approximately the same numerical value.

III. HNGA

In this section, we propose the Hybrid Niching Genetic Algorithm (HNGA), which can be used to optimize the WSVF to automatically evolve the proper number of clusters as well as appropriate partitioning of the data set. In the literature, there have been many attempts to use GAs for data clustering. In general, these attempts fall into one of the two categories. Most genetic clustering methods generally optimize the k-means or fuzzy c-means objective function [20], [28], [44]. Accordingly, these approaches assumed that the number of clusters is known *a priori*. Others developed without such assumption are to optimize validity functions [14], [34], [35]. In both categories, methods employ either the SGA or their variants, which may suffer from premature convergence. Furthermore, they may take a large amount of time to converge. In the HNGA, we develop a niching method for clustering. Additionally, in order to improve the computational efficiency, we hybridize the niching method with the computationally attractive k-means. In the following sections, we describe how the clustering solutions are initially created, how they evolve during the optimization process, how the niching method and k-means hybridization works, and how the process terminates.

A. Representation and Initialization

In most of the genetic clustering applications, binary and real parameter representations [8], [43], [47] are commonly used. In the binary representation, a binary string is usually used to represent the membership or permutation of data objects, and cluster assignment is done explicitly based on the value of the binary string. In most cases, binary representation suffers from the problems of redundancy and context insensitivity with traditional crossover and mutation operators [15]. Furthermore, they

are not a scalable representation for clustering large data sets due to the length of the chromosomes. In this paper, we use a real parameter representation, which represents the cluster centers. Cluster assignment is done implicitly based on distance. In this context, Michalewicz [36] showed that a real parameter representation moves the problem closer to the problem representation and offers higher precision with more consistent results across replications.

Our representation for individual i consists of a vector of $k_i \times d$ real numbers, where d is the number of dimensions in the data, and k_i is the number of clusters encoded in the individual. The first d positions represent the centroid of first cluster, the next d positions represent that of the second cluster, and so on. For example, in two-dimensional space, the individual's chromosome (1.56, 2.11, 23.20, 20.90, 7.85, 6.99) encodes three cluster centers (1.56, 2.11), (23.20, 20.90), and (7.85, 6.99). Each individual i in the population is constructed by random assignment of real numbers to each of the d attributes of the k_i cluster centers. The initial values are constrained to be in the range (determined from the data set) of the attribute to which they are assigned but are otherwise random. The initial cluster number k_i is calculated according to $(\text{RandInt}\ ()\ mod K^{\max}) + 2$. Here, RandInt() is a function returning a natural number, and $K^{\max}$ is the upper bound of the number of clusters and is taken to be $\sqrt{n}$, which is a rule of thumb used by many investigators in the literature [40]. The number of clusters in the population will therefore range from 2 to $K^{\max} + 1$.

B. Niching Method

Traditional GAs with elitist selection are suitable for locating the optimum of unimodal functions as they converge to a single solution of the search space. Real optimization problems such as clustering, however, often lead to many local optima. In such case, traditional GAs cannot maintain controlled competitions among the competing niches corresponding to different peaks and cause the population to converge to one alternative or another [42]. One of the key features to find the optimum of a difficult optimization problem with genetic algorithm approach is the preservation of the population diversity during the search. Diversity prevents GAs being trapped by local optima or converging prematurely. Therefore, various procedures [11], [17], [33], [41] have been developed to reduce the effect of genetic drift in the traditional GAs. They maintain the population diversity and permit the GAs to investigate many peaks in parallel. On the other hand, they prevent the GAs from being trapped in local optima of the search space.

In general, niching methods can be classified into two groups. The first one [17], [41] involves GAs, which are characterized by an explicit neighborhood since they need an explicit distance cutoff to induce emergence of niches in the search space. This feature restricts their application to clustering problems for which distance between optima cannot be estimated. The second one [11], [33] consists of techniques for which neighborhood is implicit and requires no information about the search space. A general niching technique of this category, known as deterministic crowding, is shown below and followed by a new niching method proposed for the clustering problem.

1) Deterministic Crowding: Mahfoud [33] improved the standard crowding of DeJong [11] and proposed Deterministic Crowding (DC) by introducing competition between offspring and parents of identical niches. In DC, selection pressure at the selection stage is eliminated by allowing individuals to mate at random with any other individual in the population. After crossover and eventually mutation, each of the two offspring is first paired with one of the parents; this pairing is not done randomly. Rather, the pairings are done in such a manner the offspring is paired with the most similar parent. Then, each offspring is compared with its paired parent, the individual with the higher fitness is allowed to stay in the population, and the other is eliminated.

2) Proposed Niching Method: DC can maintain the diversity of individuals in proportion to their fitness. However, in the case of our clustering problem, it is also important to maintain the population diversity with respect to the number of clusters encoded in the individuals. Therefore, a new niching method is developed to preserve both the diversity of the population with respect to the number of clusters encoded in the individuals and the diversity of the subpopulation with the same number of clusters. In this method, the selection step is modified to encourage mating within individuals with similar number of clusters and the replacement step is modified to encourage replacement within similar individuals (determined by the Euclidean distance based on a *phenotypic metric*) of the same number of clusters while allowing for some competitions among the subpopulations with different number of clusters. The similarity measure used during the selection step is the absolute difference of the number of clusters encoded in the individuals. If this result in a group with more than one candidate for selection, then the Euclidean distance based on a phenotypic metric is further used within the group. The flow of the niching method is as follows:

Step 0) (Initialization) Generate an initial population with P individuals.

Step 1) (Restricted mating) One parent p_1 is selected randomly from the population, and its mate p_2 is selected not from the entire population but from a group of individuals called *Selecting Factor Group* (SFG), which is picked randomly (without replacement) from the population. The one most similar to p_1 is chosen as mate p_2. This procedure is repeated until $P/2$ parent pairs are selected.

Step 2) Crossover the parent pairs to generate offspring, then apply mutation, and run one step of k-means on the new offspring (see Section III-D).

Step 3) (Restricted competition replacement) Compare each offspring *op* with a group of individuals called the *Comparing Factor Group* (CFG), which is picked randomly (without replacement) from the population and pair with the most similar individual of the same number of clusters as *op* if exists; otherwise, pair with the individual with the lowest fitness. If the fitness of the offspring *op* is better than its paired individual, then replace the individual.

Step 4) Repeat steps 1 to 3 until the stopping criterion is met.

Crossover between two individuals with a large difference in cluster number encoded in the chromosomes often produces low performance offspring. Therefore, we introduce *Selecting Factor Group* (SFG) to promote mating between individuals with similar number of clusters. When the SFG size value equals one, it is basically a random selection. As the size increases there is more possibility of selecting a similar individual as the first parent. However, it should be small enough to allow mating between individuals with different number of clusters. The idea of using *Comparing Factor Group* (CFG) is mostly to balance competition during replacement between individuals with the same number of clusters and individuals with different number of clusters. A large CFG size will restrict the replacement among individuals with the same number of clusters. Decreasing CFG size will promote more competition between the individuals with different number of clusters. An appropriate value should be set to allow both thorough explorations of the search space and competition between individuals with different number of clusters. By using Euclidean distance based on a phenotypic metric to measure the dissimilarity of individuals with the same number of clusters during replacement, we are also trying to preserve the diversity of the subpopulation with the same number of clusters.

C. Fitness Computation

The fitness of an individual indicates the degree of suitability of the solution it represents. Any of the functions described in Section II-A can be used for this purpose. For each individual, the centers encoded in it are first extracted, and then, a partitioning is obtained by assigning the objects to a cluster corresponding to the closest center. Given the above partitioning and the number of clusters, the value of a specific validity function is computed. The fitness of an individual is then defined as a function of the corresponding validity measure.

However, it may result in unreliable solution by using a certain validity function for computing fitness because its performance may be sensitive to structures of different problem instances. A possible way to overcome this problem is to apply the WSVF for computing the fitness of the individuals. Unlike a single validity function, the use of the WSVF is not so straightforward, because no *a priori* information available for normalization of its component functions. Here, we design our algorithm to normalize the values of its component functions dynamically during evolution. As a result the fitness of the individuals is also calculated dynamically based on the normalized values.

For each individual, a fitness vector is introduced to store the values calculated from each of the six component functions of the WSVF that are associated with the individual. Before calculating the fitness of an individual, the values of corresponding component functions in the associated fitness vector are first extracted and each of them then normalized according to

$$f'_i(x) = \frac{f_i(x) - f_i^{\min}(x)}{f_i^{\max}(x) - f_i(x)} \tag{8}$$

where $f_i(x)$ is the value for the ith component function, and $f_i^{\max}(x)$ and $f_i^{\min}(x)$ are maximum and minimum values of the component function recorded so far in the evolution. After that, the fitness of the individual is then calculated according to (7) based on the normalized values. Note that the fitness calculating of individuals happens only when they compete with their paired individuals to survive and the values will not associate with the individuals.

D. K-Means Hybridization

K-means [32] is an iterative scheme attempting to minimize the sum of squared Euclidean distances between data objects and cluster centers. Let x_i $(i = 1, 2, \ldots n)$ be the set of n data objects, K the number of clusters, and z_j the center of cluster C_j. Then, the algorithm tries to minimize the cost function Mean Square Error (MSE)

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1, x_i \in C_j}^{K} \|x_i - z_j\|^2. \tag{9}$$

Starting from an initial distribution of cluster centers in the data space, each data point is assigned to the cluster with the closest center, after which, each center itself is updated as the center of mass of all data objects belonging to that particular cluster. The procedure is repeated until convergence. This iterative scheme is known to converge sufficiently fast. However, it depends highly on the initialization of cluster centers.

In order to improve computational efficiency, one step of k-means is applied to all new offspring during each generation after the regeneration step. This is done by assigning each data point to one of the clusters with the nearest center encoded in the individual's chromosome. After that the cluster centers encoded in the individual's chromosome are replaced by the mean objects of the respective clusters.

E. Crossover and Mutation

During crossover, the cluster centers are considered to be indivisible, i.e., the crossover points can only lie in between two clusters centers. For this purpose, the operator analogous to traditional two-point crossover [18] is defined as follows: Let parent individuals P_1 and P_2 encode c_1 and c_2 cluster centers $(c_1 \leq c_2)$, respectively, and x_1 and x_2, which are the crossover points in P_1, are generated as $x_1 = \text{RandInt}(0, c_1)$ and $x_2 = \text{RandInt}(0, c_1)$. If x_1 is greater than x_2, then swap the value of x_1 and x_2 to make sure that $x_2 > x_1$. The cross points x_3 and x_4 in P_2 are then generated as $x_3 = \text{RandInt}(0, c_2 - |x_2 - x_1|)$ and $x_4 = x_3 + |x_2 - x_1|$, where $|x_2 - x_1|$ is the length of segment between cross points of x_2 and x_1. After that, the segment information between x_1 and x_2 in P_1 exchanges with the segment information between x_3 and x_4 in P_2. It can be seen that according to the above rules, the number of clusters of the offspring generated will be either have the same number of clusters as P_1 or P_2. The crossover is performed on each paired parents.

After crossover, a low probability of Gaussian mutation was applied to the offspring. Gaussian mutation adds a unit Gaussian distributed random value to the chosen attribute value. The new

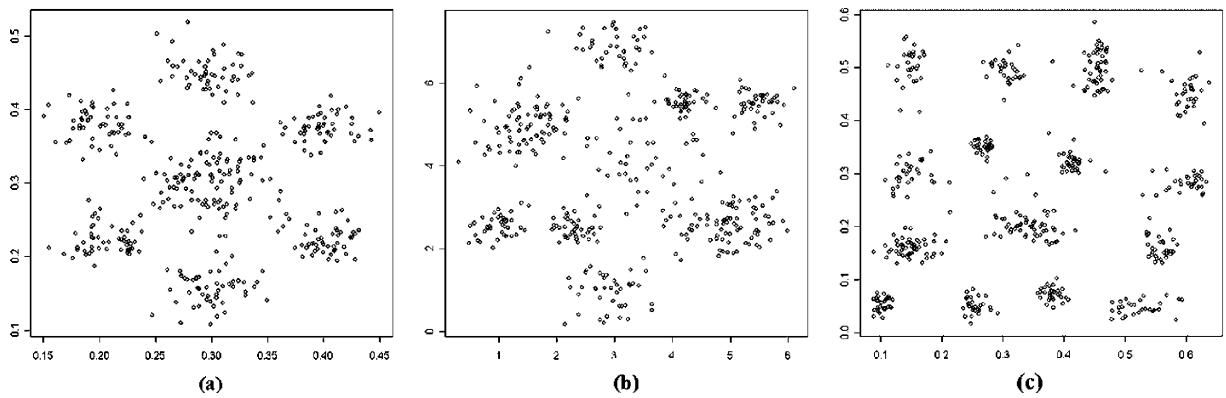

Fig. 1. (a) *Normal_7* data set. (b) *Normal_9* data set. (c) *Normal_15* data set.

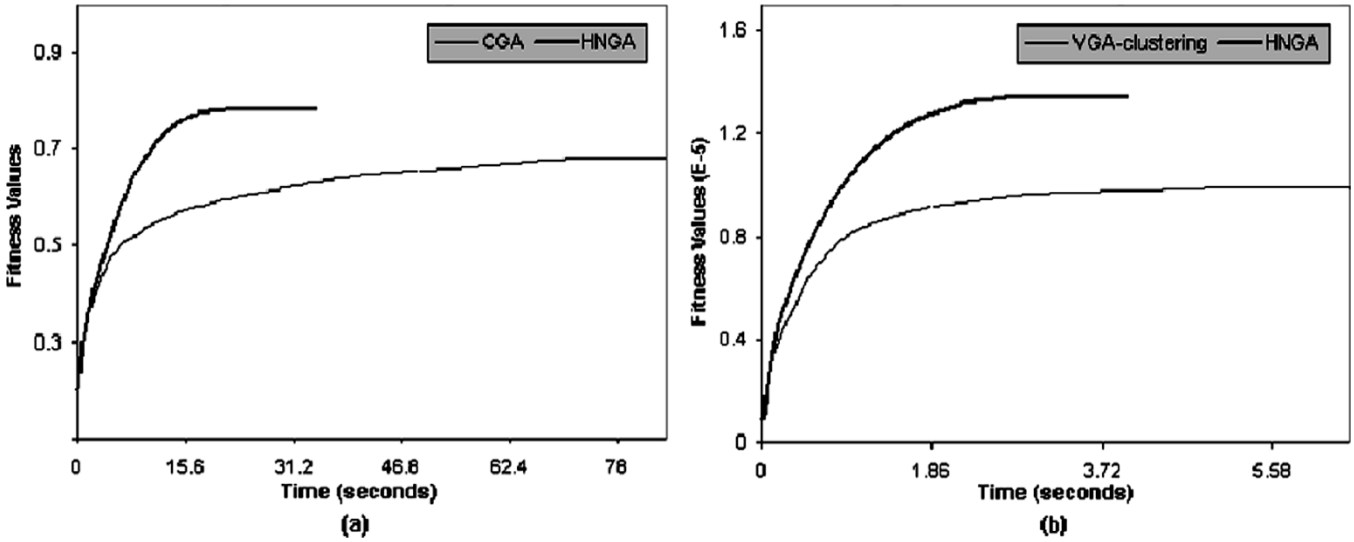

Fig. 2. (a) Fitness value (averaged over ten trials) found on *Normal_9* data set plotted against run time by the CGA and HNGA and (b) by the VGA-clustering and HNGA.

attribute value is clipped if it falls outside of the lower or upper bounds of that attribute.

F. Terminal Condition

The stopping criterion for the proposed algorithm is that the fitness value of the best population individual has not changed for n generations. There are several other possibilities for determining when to stop. Options include examining the standard deviation of the population individuals' fitness values or stopping when the current standard deviation reaches some percentage of the original population standard deviation. However, these options may be not appropriate as the stopping criteria for the HNGA since the population diversity should be preserved during the search.

IV. Data Sets and Implementation Parameters

This section provides a description of the data sets and the implementation parameters of the proposed algorithm. Several numerical data sets and real data are used in our experiments. The numerical data sets *Normal_7*, *Normal_9*, and *Normal_15* are generated according to a spherical bivariate normal distribution with 7, 9, and 15 clusters, respectively. Fig. 1(a)–(c) shows the three data sets. Note that the artificial data sets are generated in such a way that they present different degrees of difficulty for clustering. The *Normal_7* appears to be visually simple: one larger cluster in the middle rounded with six clusters with the same volume. On the other hand, clusters in *Normal_9* can be seen to be overlapped and with different volumes and sizes. In *Normal_15*, there is large number of clusters, and we also introduce some noisy objects between clusters from a uniform distribution.

Three real data sets considered are *Iris*, *Breast cancer*, and *Subcellcycle*. First, we consider Iris data [1]: The $n = 150$ objects in four dimensions represents three physical clusters Setosa, Versicolor, and Virginica, with 50 objects per cluster. It is known that two clusters (Versicolor and Virginica) have a large amount of overlap, whereas the cluster Setosa is linearly separable from the other two. Second, we will use Wisconsin breast cancer data, which is available at the UCI Repository [38]. It is a nine-dimensional data with 699 data objects. We remove 16 objects with missing values and use the remaining ones since the current version of our algorithm cannot cope with missing values. The third data set *Subcellcycle* is a subset of the yeast cell cycle data set provided by [7]. The yeast cell cycle data set contains time series expression profiles for more than 6220

TABLE I
COMPARING THE RESULTS FOUND BY THE CGA AND HNGA ON THE EXPERIMENTAL DATA SETS OUT OF TEN TRIALS

Data sets	CGA						HNGA					
	# clusters found	Count	Average fitness	Overall average fitness	Sd. dv	Run time (secondes)	# clusters found	Count	Average fitness	Overall average fitness	Sd. dv	Run time (seconds)
Normal_7	2	2	0.48	0.63	0.125	64.50	7	10	0.81	0.81	0	27.90
	3	2	0.52									
	5	1	0.62									
	6	2	0.72									
	7	1	0.81									
	9	1	0.75									
	10	1	0.72									
Normal_9	3	1	0.51	0.68	0.072	83.20	9	10	0.78	0.78	0	35.30
	4	2	0.65									
	5	2	0.67									
	6	2	0.70									
	8	2	0.73									
	9	1	0.78									
Normal_15	6	3	0.47	0.57	0.137	109.45	16	8	0.80	0.80	0.004	46.22
	8	2	0.51				17	2	0.79			
	10	2	0.50									
	13	1	0.75									
	14	1	0.76									
	17	1	0.79									
Iris	2	6	0.85	0.81	0.057	10.20	2	10	0.85	0.85	0	6.30
	3	4	0.74									
Breast cancer	2	10	0.76	0.76	0	593.46	2	10	0.76	0.76	0	382.28
Subcellcycle	2	3	0.45	0.48	0.035	1057.30	4	7	0.52	0.52	0.005	591.60
	3	3	0.50				5	3	0.51			
	4	2	0.52									
	5	1	0.46									
	6	1	0.44									

TABLE II
COMPARING THE RESULTS FOUND BY THE VGA-CLUSTERING AND HNGA ON THE EXPERIMENTAL DATA SETS OUT OF TEN TRIALS

Data sets	VGA-clustering						HNGA					
	# clusters found	Count	Average fitness	Overall average fitness	Sd. dv	Run time (secondes)	# clusters found	Count	Average fitness	Overall average fitness	Sd. dv	Run time (seconds)
Normal_7	2	3	0.48 E-5	1.01E-5	0.668	5.30	7	2	1.91 E-5	2.30 E-5	0.207	3.28
	4	2	0.63 E-5				8	8	2.40 E-5			
	6	2	0.85 E-5									
	7	1	1.91 E-5									
	8	1	2.40 E-5									
	9	1	1.31 E-5									
Normal_9	2	1	0.34 E-5	0.98 E-5	0.392	6.45	6	10	1.35 E-5	1.35 E-5	0	4.20
	3	1	0.50 E-5									
	4	1	0.65 E-5									
	5	1	0.64 E-5									
	6	4	1.30 E-5									
	7	2	1.21 E-5									
Normal_15	5	1	0.52 E-5	0.58 E-5	0.048	6.53	8	10	0.67 E-5	0.67 E-5	0.021	4.41
	6	2	0.57 E-5									
	7	3	0.61 E-5									
	8	1	0.64 E-5									
	9	1	0.62 E-5									
	11	2	0.56 E-5									
Iris	2	3	0.65 E-3	1.04 E-3	0.298	3.53	3	10	1.32 E-3	1.32 E-3	0	2.81
	3	3	1.32 E-3									
	4	3	1.20 E-3									
	5	1	0.86 E-3									
Breast cancer	2	10	6.30 E-6	6.30 E-6	0.071	13.80	2	10	6.35 E-6	6.35 E-6	0	7.96
Subcellcycle	2	2	1.39 E-6	1.08 E-6	0.221	10.30	2	10	1.39 E-6	1.39 E-6	0.012	7.44
	3	3	1.02 E-6									
	4	3	0.92 E-6									
	5	1	1.19 E-6									
	6	1	0.94 E-6									

genes, with 17 time points for each gene taken at 10-min intervals covering nearly two yeast cell cycles (160 min). The *Subcellcycle* data used in this work consists of 384 genes, whose expression levels peak at different time points corresponding to the five phases (early G1, late G1, S, G2 and M) of cell cycle. The *Subcellcycle* is normalized so that every gene has an average expression value of zero and a standard deviation equal to one.

All parameter values of the HNGA for experiments on above data sets were determined experimentally. The mutation rate was set at 0.01. To establish this value, all variables were held constant with only the mutation rate changing. Ten runs were

completed for a wide range of values of the mutation rate. The best solutions from each of the ten runs were averaged, and the best average was selected. The outcomes of the ten runs were averaged to ensure the HNGA performs well consistently over a series of runs. The size of SFG and CFG were determined in a similar way. First, SFG size was varied as all other parameters were held constant, and then, the CFG size was determined using the established SFG size. Both the best fitness values and consistency of results were used in determining the SFG and CFG size. The SFG size of 4, 5, 5, 3, 6, and 5 with the CFG size of 10, 15, 15, 8, 20, and 15 were established on the above six data sets, respectively. The population size was $3 \times K^{\max}$ for each data set so that large data set will have large population size [45]. The stopping criterion was that the fitness value of the best population individual has not changed for 20 generations. It is possible that there are nonlinear relationships between the parameters and therefore, there may be a more effective set of parameter values that would result in even better performance.

V. Experimental Results

In order to evaluate the proposed method, we performed two types of experiments. First, we explored the properties of the HNGA and compared the performance with other genetic clustering methods, which do not need knowledge about the number of clusters *a priori*. After that, we compared the performance of the WSVF with its six component functions as clustering objective function. All results reported in this section were obtained on a PC with AMD Athlon™ 1800 running the Windows™ 2000 operation system.

A. Exploring the HNGA

In this section, we explored the properties of the HNGA and compared it with the Clustering Genetic Algorithm (CGA) [14] and the VGA-clustering [34]. Other than the quality of clustering solutions, which is also dependent on the specific clustering objective function for computing fitness, we mainly concern the performance with respect to the criteria of the *consistency* and *efficiency* of identifying the best known optimum corresponding to the given data in concurrence with the convergence results.

Before discussing the comparative experiments, we first briefly describe the CGA and the VGA-clustering. The CGA was developed as an attempt to improve Falkenauer's Grouping Genetic Algorithm [15]. In the CGA, the binary representation was used to represent the membership of data objects, and cluster assignment was done explicitly based on the value of the binary string. To avoid the problems of redundancy and context insensitivity, both crossover and mutation operators in the CGA were redefined. In the VGA-clustering, the Variable string length Genetic Algorithm (VGA) [19] was applied with real parameter representation as the underlying searching tool. Both the CGA and the VGA-clustering employed a variant of SGA, in which individuals with higher fitness scores are selected to create a mating pool, and the next generation is comprised of those previous generation individuals who never got a chance to reproduce and the offspring of those who got a change to produce.

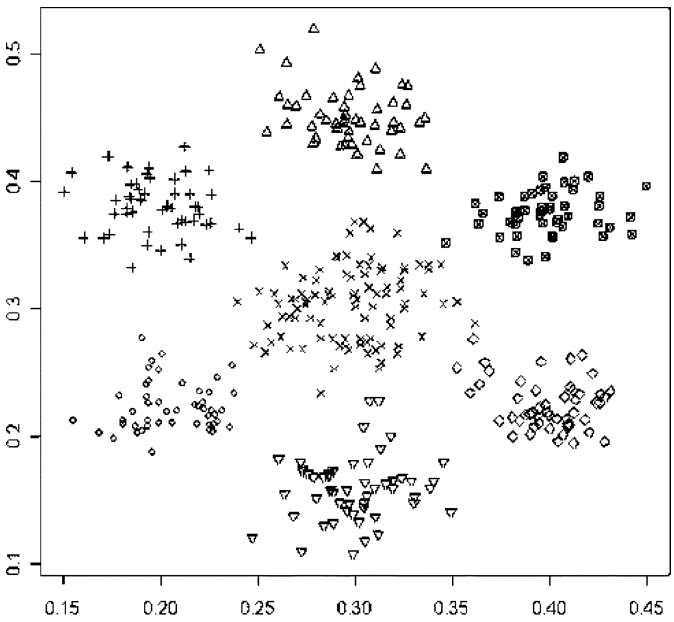

Fig. 3. Partitioning with seven clusters found on *Normal_7* data when the WSVF is used for computing fitness.

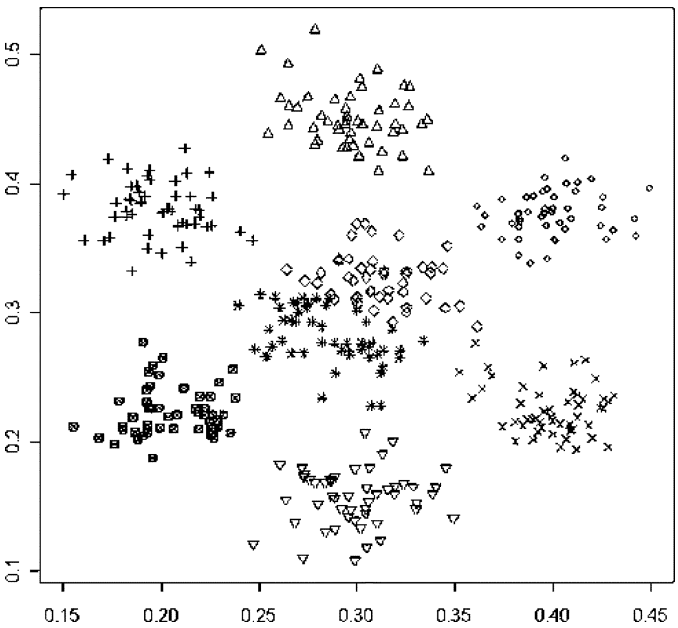

Fig. 4. Partitioning with eight clusters found on *Normal_7* data when the PBM is used for computing fitness.

Since different functions were used in the CGA and the VGA-clustering for computing fitness, we carry out two pairwise comparisons, between the HNGA and the CGA and between the HNGA and the VGA-clustering. In the first pairwise comparison, both the HNGA and the CGA run on the experimental data sets to optimize the SIL, which was used for computing fitness in the CGA, and their results were compared. Similarly, the second pairwise comparison between the HNGA and the VGA-clustering was carried out, however, to optimize the PBM, which was used for computing fitness in the VGA-clustering. To make the pairwise comparisons more meaningful, we used the same stopping criterion (i.e., the fitness value of the best population individual has not changed for 20 generations) and the

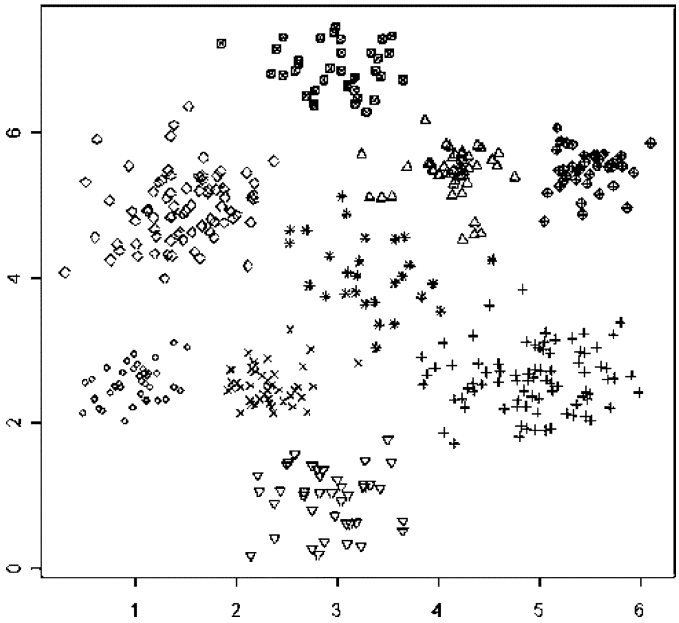

Fig. 5. Partitioning with nine clusters found on *Normal_9* data when the WSVF is used for computing fitness.

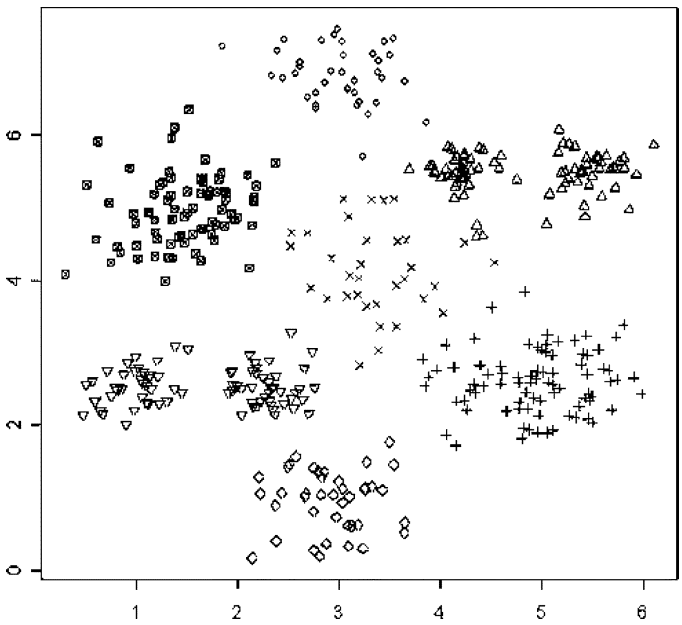

Fig. 6. Partitioning with seven clusters found on *Normal_9* data when the V_D is used for computing fitness.

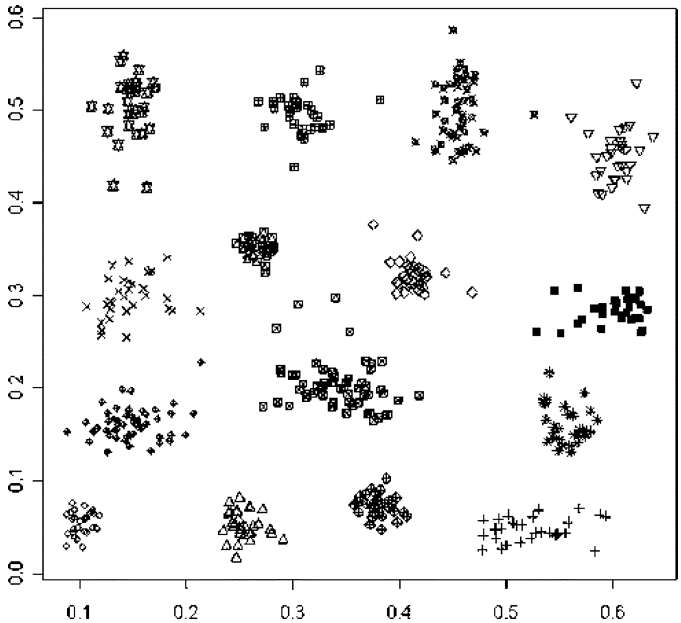

Fig. 7. Partitioning with fifteen clusters found on *Normal_15* data when the WSVF is used for computing fitness.

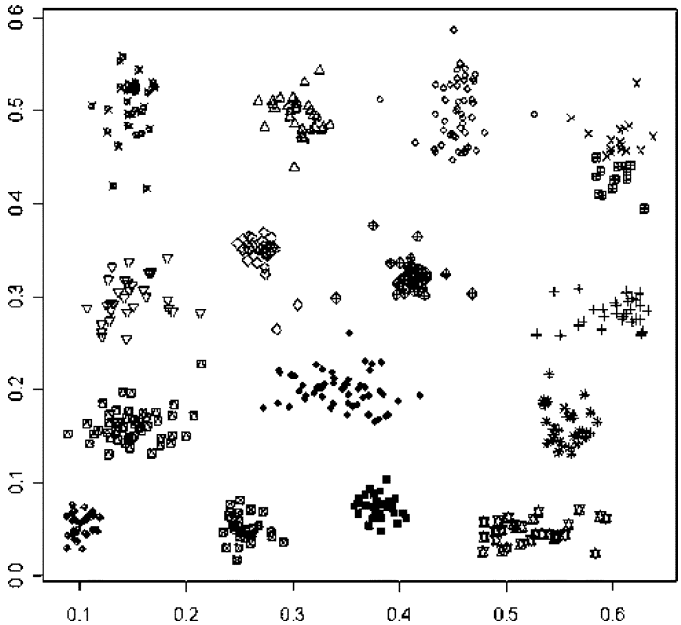

Fig. 8. Partitioning with sixteen clusters found on *Normal_15* data when DB is used for computing fitness.

same population size on each data set for all experiments. Other parameters of the CGA and the VGA-clustering were specified according to the original papers for the best performance.

To compare the performance of the two pairwise comparisons in terms of the consistency and efficiency of identifying the best known optimum corresponding to the given data in concurrence with the convergence results, we reported the *number of clusters*, *count*, *average fitness value*, *overall average fitness value*, and *average run time* found from the ten trials on each of the six data sets. The number of clusters is the resulting cluster number found from the trials, count is the number of trials for which the resulting cluster number occurred, the average fitness is averaged over the count, the overall average fitness and average run time is averaged over all of the ten trials. The ten trials were carried out by generating ten random starting populations,and running each experiment once with each of the ten random starting populations. All sets of experiments used the same ten random starting populations. The same starting populations were used to better ensure that the different performance was not caused by different starting populations. Tables I and II show these two pairwise comparison results on the six data sets. The performance of the two pairwise comparisons on *Normal_9* data in

TABLE III
COMPARING THE NUMBER OF CLUSTERS FOUND ON THE EXPERIMENTAL DATA SETS OUT OF TEN TRIALS BY USING THE WSVF AND EACH OF ITS COMPONENT FUNCTIONS AS THE CLUSTERING OBJECTIVE FUNCTION FOR COMPUTING FITNESS. IF THERE ARE A DIFFERENT NUMBER OF CLUSTERS IDENTIFIED OUT OF TEN TRIALS, THE NUMBER OF TRIALS OCCURRING ON EACH OF THE NUMBER OF CLUSTERS IS INDICATED IN THE PARENTHESES

Data sets	Actual # clusters	Fitness Function						
		1/DB	CH	PBM	SIL	V_D	V_{33}	WSVF
Normal_7	**7**	7(7) 8(3)	7	7(2) 8(8)	7	5	7	7
Normal_9	**9**	9	11	6	9	7	7(8) 8(2)	9
Normal_15	**15**	16	15	8	16(8) 17(2)	15	10	15
Iris	**2 or 3**	2	3	2	2	2	2	2
Breast cancer	**2**	2	2	2	2	2	2	2
Subcellcycle	**5**	5	3	2	4(7) 5(3)	5	3	5

terms of the fitness value against run time is shown in Fig. 2(a) and (b), respectively.

Tables I and II show that almost every run of the HNGA eventually converges to the best known optimum value on all experimental data sets, whereas most runs of the CGA and the VGA-clustering fail to do so. The performance of the CGA and the VGA-clustering is not surprising because they are suffered from premature convergence to local optima, e.g., nine out of ten trials of both the CGA and the VGA-clustering on *Normal_7* data are trapped in local optima. Similar results can be also observed on *Normal_9* data from Fig. 2(a) and (b) that, in case of the HNGA, the average fitness values is approaching the best known one, whereas it is not in the case of the CGA and the VGA-clustering. Therefore, from the tables and graph, we can infer that the CGA and the VGA-clustering are not assured to reach the best known optimum value. The situation becomes worse when the search space is large, and there are many local optima, e.g., neither CGA nor VGA-clustering can identify the best known optimum on *Normal_15* data out of ten trials. Furthermore, the CGA and the VGA-clustering are not efficient enough. However, the HNGA is faster to identify the best known optimum value on all experimental data sets. This mainly comes from the one step of the k-means hybrid. Clearly, based on the experiments of the two pairwise comparisons, the HNGA is the best alternative as it can consistently and efficiently converge to the best known optimum corresponding to the given data in concurrence with the convergence results.

It was also observed, in the experiments, that even though with the best known fitness value, the resulting number of clusters may not correspond to the correct one. For example, the best known fitness value $f = 0.80$ (the SIL was used for computing fitness) and $f = 0.67\text{E-}5$ (the PBM was used for computing fitness) on *Normal_15* data correspond to 16 and eight clusters, respectively. This indicates that the SIL and PBM may have limitations as clustering objective functions. This problem will be further explored in the following section.

B. Exploring the WSVF

In the next experiments, we explored the properties of the proposed WSVF as the clustering objective function and compared it with its six component functions. We compared the results with respect to the criteria of 1) the *accuracy* to recover clusters in the data sets and 2) the *robustness* over all the experimental data sets. Since the performance of the specific function is also dependent on the genetic clustering scheme, the same setting of the HNGA is used for the optimization of the different functions as the clustering objective to make the comparisons more meaningful.

The number of clusters obtained by using each of the six validity functions together with the WSVF are shown in Table III. As in previous experiments, the results were obtained by using the same ten random starting populations and running each experiment once with each of the ten random starting populations. Figs. 3, 5, and 7 show the partitioning obtained by using the WSVF on the three artificial data sets (different clusters represented by different symbols). As a comparison, we also show one typical partitioning obtained by using the component functions of WSVF on each of the three artificial data sets in Figs. 4, 6, and 8, respectively.

It can be seen from Table III that the WSVF can be used to accurately determine the number of clusters in a data set and is robust as it gave uniformly good results over all the data sets. Figs. 3, 5, and 7 show that the WSVF can evolve the appropriate partitioning for all the three artificial data sets. The six component functions failed at least one of the six data sets considered. The $\mathrm{V_D}$ failed on two data sets (*Normal_7* and *Normal_9*), and the $\mathrm{V_{33}}$ failed on three data sets (*Nomal_9*, *Normal_15*, and *Subcellcycle*). Neither the $\mathrm{V_D}$ nor $\mathrm{V_{33}}$ were able to identify the presence of the nine overlapping clusters with different volumes and sizes in *Normal_9* data. We can see, from Fig. 6, that $\mathrm{V_D}$ found a partitioning with seven clusters on *Normal_9* as two clusters at the lower-left corner and two clusters at up-right corner are joined together. In the simulations from *Normal_9*, the CH and PBM also failed to recover the proper number of clusters, whereas the DB and SIL performed well. However, for *Normal_15* with large number of clusters and some noise objects, the DB and SIL cannot recover the proper number of clusters, whereas the CH and $\mathrm{V_D}$ showed the best performance. Fig. 8 shows that DB preferred a partitioning with 16 clusters on *Normal_15*, as the cluster at the upper right corner is split up into two clusters.

For *Subcellcycle* data, apart from the WSVF, only the DB, $\mathrm{V_D}$ and three out of ten trials of the SIL provided the correct number of clusters, whereas others tended to underestimate the

number of clusters. For Iris data, all functions show that $K = 2$ is the optimal number of clusters, except the CH. Geometrically, the primary structure in Iris is probably $K = 2$, but the physical labels insist that $K = 3$. Consequently, the best value for K is debatable. Breast cancer data has two clusters, which have only a small amount of overlap. As a result, two clusters were found irrespective of the function used.

In summary, for the data sets considered here, the WSVF is the most robust and accurate to recover the proper number of clusters and the appropriate partitioning. The WSVF can integrate many functions to yield stable results. Thereby, the user does not have to compare different functions and pick a single best function. Rather, the WSVF appears to automatically "focus" on whatever is most appropriate for the given data.

VI. Summary and Discussion

In this paper, we consider the clustering problems wherein the number of clusters is not known *a priori*. We suggest a clustering objective function called Weighted Sum Validity Function (WSVF), which is a weighted sum of the several normalized cluster validity functions. Further, we propose a Hybrid Niching Genetic Algorithm (HNGA) that can be used for the optimization of the WSVF to automatically evolve the proper number of clusters and appropriate partitioning of the data set. Within the HNGA, a niching method is developed to prevent premature convergence during the search. Additionally, in order to improve the computational efficiency, we hybridize the niching method with the computationally attractive k-means. In comparison with other related genetic clustering algorithms, the resulting HNGA can *consistently* and *efficiently* converge to the best-known optimum corresponding to the given data in concurrence with the convergence results. The WSVF is found to generally be able to improve the confidence of clustering solutions and achieve more *accurate* and *robust* results.

The HNGA clustering by optimizing the WSVF requires more computational time than general clustering methods such as k-means. This is mainly because the WSVF consists of six validity functions to evaluate the given solution chromosome. For example, it takes two orders of magnitude more time than k-means for the Iris data set. However, the reward is that our approach can consistently achieve accurate and robust results, without making any *a priori* assumption about the number of clusters. Like most other general clustering methods, k-means needs knowledge about the number of clusters *a priori* and is susceptible to local optima. To make the comparison between the HNGA clustering by optimizing the WSVF and k-means more meaningful, we may need to process k-means with $K = 2$ to $K = K^{\max}$, where $K^{\max}$ is taken to be $\sqrt{n}$, and evaluate the WSVF of the resulting partitioning in each case. Then, this procedure is repeated until the same amount of time as the HNGA clustering is spent, and the best partitioning is selected. In this case, our results indicate (data not shown) that the HNGA clustering by optimizing the WSVF outperformed k-means in terms of the WSVF values with only one exception, i.e., for the Breast cancer data, both methods performed equally well. Similarly, the comparisons between the HNGA clustering by optimizing the WSVF and other general clustering methods may also be carried out in the future.

The HNGA clustering by optimizing the WSVF becomes costly for the larger sizes of data sets, which leads to a quadratic increase in the computation time. To alleviate this drawback, one may use fewer component functions for the WSVF and/or run the method on a subset of the data set (e.g., 10%) selected using sampling techniques.

There are several other directions in which the work may be extended further. First, how to effectively choose the component functions for the WSVF to achieve even better results is still an open problem. The HNGA can preserve the subpopulations with a different number of clusters during the search. Accordingly, parallel computer architectures can be exploited. Additionally, simultaneous search for the effective set of parameter values of the developed technique can be attempted, and the use of other distance metrics may be investigated. Finally, the algorithm developed here can be suitably modified and tailored so that it is applicable to fuzzy clustering problems. In this regard, a clustering objective function consisting of several fuzzy cluster validity functions may also be developed in a similar way as the WSVF.

References

[1] E. Anderson, "The Irises of the gaspe peninsula," *Bull. Amer. Iris Soc.*, vol. 59, pp. 2–5, 1935.

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

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

[4] J. C. Bezdek and N. R. Pal, "Some new indexes of cluster validity," *IEEE Trans. Syst., Man, Cybern.*, pt. B, vol. 28, no. 3, pp. 301–315, Jun. 1998.

[5] J. C. Bezdek, J. Keller, R. Krisnapuram, and N. R. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Boston, MA: Kluwer, 1999.

[6] R. B. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Statistics*, vol. 3, pp. 1–27, 1974.

[7] R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrieflian, D. Landsman, D. J. Lockhart, and R. W. Davis, "A genome-wide transcriptional analysis of the mitotic cell cycle," *Molecular Cell*, vol. 2, no. 1, pp. 65–73, 1998.

[8] R. Cucchiara, "Genetic algorithms for clustering in machine vision," *Machine Vision Applicat.*, vol. 11, no. 1, pp. 1–6, 1998.

[9] R. N. Dave and T. Fu, "Robust shape detection using fuzzy clustering: Practical applications," *Fuzzy Sets Syst.*, vol. 65, pp. 161–185, Jan. 1995.

[10] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI–1, pp. 224–227, 1979.

[11] K. A. DeJong, "An Analysis of the Behavior of a Class of Genetic Adaptative Systems," Ph.D. dissertation, Univ. Michigan, Ann Arbor, MI, 1975.

[12] R. Dubes and A. K. Jain, "Clustering techniques: The user's dilemma," *Pattern Recogn.*, vol. 8, pp. 247–260, 1976.

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

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

[15] E. Falkenauer, *Genetic Algorithms and Grouping Problems*. New York: Wiley, 1998.

[16] I. Gath and G. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Machin. Intell.*, vol. 11, pp. 773–781, Jul. 1989.

[17] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. Second Int. Conf. Genetic Algorithms*, 1987, pp. 41–49.

[18] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[19] D. E. Goldberg, K. Deb, and B. Korb, "Messy genetic algorithms: Motivation, analysis, and first results," *Complex Syst.*, vol. 3, pp. 493–530, 1989.
[20] L. O. Hall, I. B. Ozyurt, and J. C. Bezdek, "Clustering with a genetically optimized approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 103–112, Jul. 1999.
[21] J. A. Hartigan, *Clustering Algorithms*. New York: Wiley, 1975.
[22] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[23] F. Hopper, F. Klawonn, R. Kruse, and T. Runkler, *Fuzzy Cluster Analysis*. New York: Wiley, 1999.
[24] L. J. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, pp. 193–218, 1985.
[25] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.
[26] J. M. Jolion, P. Meer, and S. Bataouche, "Robust clustering with applications in computer vision," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 8, pp. 791–802, Aug. 1991.
[27] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: Wiley, 1990.
[28] K. Krishna and M. N. Murty, "Genetic k-means algorithm," *IEEE Trans. Syst., Man, Cybern. B*, vol. 29, no. 3, pp. 433–439, Jun. 1999.
[29] R. Krishnapuram and C. P. Freg, "Fitting an unknown number of lines and planes to image data through compatible cluster merging," *Pattern Recogn.*, vol. 25, no. 4, pp. 385–400, 1992.
[30] R. Krishnapuram, H. Frigui, and O. Nasraoui, "Fuzzy and possibilistic shell clustering algorithms and their application to boundary detection and surface approximation," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 1, pp. 29–60, Feb. 1995.
[31] G. N. Lance and W. T. Williams, "A general theory of classificatory sorting strategies: II clustering systems," *Comput. J.*, vol. 10, pp. 271–277, 1967.
[32] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Mathematical Statist. Probability*, 1967, pp. 281–97.
[33] S. W. Mahfoud, "Niching Methods for Genetic Algorithms," Ph.D. dissewrtation, Univ. Illinois, Urbana, IL, 1995.
[34] U. Maulik and S. Bandyopadhyay, "Non-parametric genetic clustering: Comparison of validity indices," *IEEE Trans. Syst., Man, Cybern. B*, vol. 31, no. 1, pp. 120–125, Feb. 2001.
[35] ——, "Genetic clustering for automatic evolution of clusters and application to image classification," *Pattern Recogn.*, vol. 35, pp. 1197–1208, 2002.
[36] Z. Michalewicz, *Genetic Algorithms + Data Structure = Evolution Programs*, Third ed. New York: Springer-Verlag, 1996.
[37] G. Milligan and M. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, pp. 159–179, 1985.
[38] P. M. Murphy and D. W. Aha. (1994) "UCI Repository for Machine Learning Databases," Tech. Rep.. Univ. Calif., Dept. Inf. Comput. Sci., Irvine, CA. [Online]. Available: http://www.ics.uci.edu/mlearn/ML-Repository.html
[39] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik, "Validity index for crisp and fuzzy clusters," *Pattern Recogn.*, vol. 37, pp. 487–501, 2004.
[40] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy c-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 370–379, Aug. 1995.
[41] A. Petrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. IEEE Int. Conf. Evol. Comput.*, Nagoya, Japan, 1996, pp. 798–803.
[42] B. Sareni and L. Krähenbühl, "Fitness sharing and niching methods revisited," *IEEE Trans. Evol. Comput.*, vol. 2, no. 3, pp. 97–106, Sep. 1998.
[43] M. Sarkar, B. Yegnanarayana, and D. Khemani, "A clustering algorithm using an evolutionary programming-based approach," *Pattern Recogn. Lett.*, vol. 18, pp. 975–986, 1997.
[44] P. Scheunders, "A genetic c-means clustering algorithm applied to color image quantization," *Pattern Recogn.*, vol. 30, no. 6, pp. 859–866, 1997.
[45] W. Sheng, A. Tucker, and X. Liu, "Clustering with niching genetic k-means algorithm," in *Proc. Genetic Evol.y Comput. Conf.*, Seattle, WA, 2004, pp. 162–173.
[46] C. V. Stewart, "MINPRAN: A new robust estimator for computer vision," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 10, pp. 925–938, Oct. 1995.
[47] G. Syswerda, "A study of reproduction in generational and steady-state genetic algorithms," in *Foundations of Genetic Algorithms*. San Francisco, CA: Morgan Kauffmann, 1991, pp. 94–101.
[48] X. Zhuang, Y. Huang, K. Palaniappan, and Y. Zhao, "Gaussian mixture density modeling, decomposition and applications," *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1293–1302, Sep. 1996.

Weiguo Sheng received the M.Sc. degree in information technology from the University of Nottingham, Nottingham, U.K., in 2002. Then, he received a studentship from Brunel University, London, U.K. and is currently working toward the Ph.D. degree in computer science with the Intelligent Data Analysis group.

His main research interests include clustering, optimization, evolutionary computation, genetic algorithm, hybrid models, and bioinformatics applications.

Stephen Swift received the B.Sc. degree in mathematics and computing from the University of Kent, Canterbury, U.K., the M.Sc. in artificial intelligence from Cranfield University, Cranfield, U.K., and the Ph.D. degree in intelligent data analysis from Birkbeck College, University of London, London, U.K.

He is currently a Research Lecturer with the School of Information Systems, Computing, and Mathematics at Brunel University, London. He has also spent four years in industry as a web designer, programmer, and technical architect. His research interests include multivariate time series analysis, heuristic search, data clustering, and evolutionary computation.

Leishi Zhang received the B.A. degree in information management from Anhui University, Anhui, China, the M.Sc. degree in applied computing from University of Dundee, Dundee, U.K., and the M.Phil. degree in bioinformatics visualization from Brunel University, London, U.K., where she is pursuing the Ph.D. degree with the Information System and Computing Department.

She is currently working on DNA microarray data analysis and visualization with the Intelligent Data Analysis group. Her research interests include computer graphics, three-dimensional visualization, and intelligent data analysis.

Xiaohui Liu is a Professor of Computing at Brunel University, London, U.K. He leads the Intelligent Data Analysis (IDA) Group, performing interdisciplinary research involving artificial intelligence, dynamic systems, image and signal processing, and statistics, particularly for biomedical and engineering applications.

Dr. Liu serves on the editorial boards of four computing journals, founded the biennial international conference series on IDA in 1995, and has given numerous invited talks in bioinformatics, data mining, and statistics conferences.