7th International Conference on Intelligent Control and Information Processing
December 1-4, 2016; Siem Reap, Cambodia

Automatic Clustering Approach Based on Particle Swarm Optimization for Data with Arbitrary Shaped Clusters

Geng-Bin Chen, An Song, Chun-Ju Zhang,
and Xiao-Fang Liu
Sun Yat-sen University
Guangzhou, China

Wei-Neng Chen*, Zhi-Hui Zhan, Jing-Hui Zhong,
and Jun Zhang
School of Computer Science and Engineering
South China University of Technology, Guangzhou, China

Xiao-Min Hu
School of Computer Science and Technology
Guangdong University of Technology, China

***Abstract*—Recently, partitional clustering approaches based on Evolutionary Algorithms (EAs) have shown promising in solving the data clustering problems. However, with the nearest prototype (NP) rule as the method for decoding, most of them are only suitable for clustering datasets with convex (e.g. hyperspherical) clusters. In this paper, we propose an automatic clustering approach using particle swarm optimization (PSO). A new encoding scheme with a novel decoding method, named the nearest multiple prototypes (NMP) rule, is applied to the PSO-based clustering algorithm to automatically determine an appropriate number of clusters in the procedure of clustering and partition datasets with arbitrary shaped clusters. The algorithm is experimentally validated on both synthetic and real datasets. The results show that the proposed PSO-based approach is very competitive when comparing with two popular clustering algorithms.**

***Keywords*—Partitional clustering, evolutionary algorithm, particle swarm optimization (PSO), prototype-based encoding, multiple prototypes**

I. INTRODUCTION

Clustering analysis, also known as unsupervised classification or exploratory data analysis, is aimed at abstracting the underlying structure of data by partitioning a finite unlabeled data set into groups or clusters of similar objects [1], [2], [3]. Clustering techniques are subdivided into hierarchical clustering and partitional clustering based on the type of structure imposed on the data [1], [2].

Partitional clustering can be regard as a non-deterministic polynomial hard (NP-hard) combinational optimization problem [4]. As the problem is complex, traditional deterministic local search algorithms, e.g. K-means [5], are easily stuck in local minima and their results are heavily influenced by the initial choice of cluster centers [6]. Therefore, more powerful search methods such as evolutionary algorithms (EAs) [7], [8] and tabu search (TS) [9], known as general-purpose metaheuristics, are applied to explore the clustering solution space more efficiently [10]-[12]. Among them, EAs have shown to be promising alternatives mainly because they have been proven to be an effective way to find solutions close to the global optimum and are less dependent upon the initial conditions. Especially, it has been shown to perform more promisingly and more efficiently than traditional randomized approaches (e.g., multiple runs of K-means) in clustering problems [13]. Recently, some clustering methods using EAs [16], [17] can evolve the optimal number of clusters in the clustering procedure without a priori knowledge of actual number of clusters, which is usually unavailable in most real-life applications. This feature makes EAs more practical and effective for clustering problems.

In partitional clustering algorithm based on EAs, there are three major representation schemes to present a clustering solution (partition): label-based, medoid-based and centroid-based representations [14], which have different decoding method from individual (genotype) to partition (phenotype). Centroid-based [16] and medoid-based [15] representations belong to prototype-based representation, which encode the prototypes, feature vectors used to represents given clusters, into a given genotype. We have concentrated on prototype-based representation for the reason that the prototype-based encoding is more scalable than label-based representation in most cases [14]. The partition can be derived by the nearest prototype (NP) rule that for a given set of cluster prototypes, any data object can be optimally classified by assigning it to the cluster whose prototype is most similar to the data object. The similarity can be measured by some criteria such as Euclidean distance.

*Corresponding Author, Email: cwnraul634@aliyun.com

This work was supported in part by the National Natural Science Foundation of China under Grant 61622206, Grant 61379061, and Grant 61332002, in part by the Natural Science Foundation of Guangdong under Grant 2015A030306024, and in part by the Guangdong Special Support Program under Grant 2014TQ01X550.

Such encoding scheme are popularly used in clustering algorithms based on EAs [14], [16]. However, these algorithms are usually biased toward the discovery of convex (e.g. hyperspherical or hyperelliptic) clusters which clearly will be inappropriate in many applications where the natural clusters for the data are non-convex.

In order to discover clusters of more complex (e.g., nonconvex) shapes and perform automatic clustering at the same time, a novel clustering approach using PSO is proposed, we refer to it as the automatic clustering PSO (ACPSO) algorithm. In this approach, we proposed a new encoding method which combines the particle representation scheme proposed by Das et al. [16], which both encodes activation thresholds and cluster centroids into particle, with a novel cluster-recovering rule as decoding method, named nearest multiple prototypes (NMP) rule. The proposed rule distinguish itself from the nearest prototype rule mainly based on three aspects as below.

1) In the NP rule, there is only one prototype in each cluster leading to the central gathering phenomenon, whilst in the NMP rule, multiple prototypes strategy is adopted to avoid this phenomenon.

2) In the NP rule, the distance between a data object and a cluster is defined as the distance of this data object to the only one prototype of the cluster, whilst in the NMP rule, corresponding to multiple prototypes strategy, a new distance measuring method is used which takes all prototypes of a cluster into account and selects the one nearest to the data object as reference substance when measuring object-cluster distance.

3) In the NP rule, each data object is assigned independently, whilst in the NMP rule, the assignment of each data object is relevant and the procedure of assignment is iterative and dynamic.

In addition to that, we propose a new population initialization method to weed out the weakness of the original particle representation scheme.

The rest of this paper is organized as follows. Section II briefly describes standard local version PSO algorithm cluster validity indexes. Section III develops the ACPSO algorithm in detail. Experimental results on fifteen synthetic and three real data sets are displayed in Section IV . Finally, conclusion is given in Section V.

II. Background

A. Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic search algorithm using a population of potential solution to search the search space [19] and [20]. The algorithm models the social behavior of a bird flock. Each particle of the population (called swarm) flies like a bird to locate an optimal location (solution) in the search space and adjusts its own velocity according to the information shared in the swarm while searching.

In the literature, there have been a lot of PSO variants proposed so far [19]-[25] when it is first introduced by Eberhart and Kennedy. In our experiments, we use the local version PSO [21]. It can be stated mathematically as follows.

Let $\mathrm{X}_i = (x_{i1}, x_{i2}, ..., x_{i2})$ and $\mathrm{V}_i = (v_{i1}, v_{i2}, ..., v_{id})$ be the position and velocity d-dimensional vectors of particle *i*. The algorithm uses the predefined fitness function, which is problem-dependent, to evaluate the position. The best previous position, representing the best function value, of the *i*-th particle is denoted as $p_i = (\rho_{i1}, \rho_{i2}, ...\rho_{id})$. The social sharing of information is based on two best locations during the evolution process: one is the particle's own previous best position, the other is the best position in the neighborhood denoted by the index *l*. Here, the neighborhood is defined by the ring topological structure. Using the above notation, the PSO velocity and position equations are calculated as

$$\mathrm{V}_{ij}(t+1) = w\mathrm{V}_{ij} + c_1 rand_1(p_i - \mathrm{X}_{ij}(t)) + c_2 rand_2(p_l - \mathrm{X}_{ij}(t)) \quad (1)$$

$$\mathrm{X}_{ij}(t+1) = \mathrm{X}_{ij}(t) + \mathrm{V}_{ij}(t+1) \quad (2)$$

where *t* is the generation, *w* is the inertia weight which play an important role of balancing global exploration and local exploitation for different problems, c_1 and c_2 are two positive acceleration constants, known as the cognitive and social components responsible for degree of information consideration of personal and swarm memory respectively; $rand_1$ and $rand_2$ are the uniformly distributed variables in the range of [0, 1]. The velocity of each particle is clamped to a maximum velocity *Vmax*. *Vmax* is often set to about 10%–20% of the dynamic range of the variable on each dimension.

B. Cluster Validity Indexes

Cluster validity indexes are statistics used to evaluate the quality of clustering structures in a quantitative and objective way [1]. Abundant indexes with different characteristics and properties exist in the literature, such as the Cali´nski-Harabasz (*CH*) index [17], the Davies-Bouldin (*DB*) index [26], the *Dunn* index [18], and the silhouette statistic (*SIL*) index [27]. Any index that is non-monotonic with the number of clusters can be potentially used as a fitness function for EAs to optimize the number of clusters and locate the corresponding best partition simultaneously. In the following, two validity measures employed in our study will be described in detail.

1) *CH* Index: The *CH* index [17] outperforms other 29 indices in Milligan and Cooper's comparative study. N data objects X which is defined as

$$CH = \frac{trace\ B/(k-1)}{trace\ W/(n-k)} \quad (3)$$

where *n* and *k* are the total number of objects and the number of clusters in the partition, respectively; *B* and *W* are the between-cluster and the pooled within-cluster sums of square (covariance) matrices, respectively. A large value of CH indicates the occurrence of the best clustering partition.

2) *Dunn* Index: The *Dunn* index [18] is designed to identify clusters that are compact and well separated. Let C_i and C_j be two nonempty clusters. The distance between *Ci* and *Cj* is defined as

TABLE I
DESCRIPTION OF THE DATA SETS AND PARAMETER SETUP OF THE DBSCAN

Dataset	Number of data objects	Number of clusters	Data dimension	ε	MinPts
Four-GussianClusters	1500	4	2	0.11	52
Three-rectangles	350	3	2	0.09	1
Two-rings	250	2	2	0.09	3
Half-rings	500	2	2	0.11	1
Ring-GaussianCluster	300	2	2	0.12	11
2d4c	1078	4	2	0.08	22
2d10c	3073	10	2	0.07	75
2d20c	1517	20	2	0.03	7
10d4c	958	4	10	0.30	35
10d10c	2729	10	10	0.21	9
10d20c	1316	20	10	0.26	20
50d4c	683	4	50	0.54	17
50d10c	2242	10	50	0.34	47
100d4c	629	4	100	0.72	9
100d10c	2103	10	100	0.50	32
Iris	150	3	4	0.09	4
Wine	178	3	13	0.37	8
Breast cancer	683	2	9	0.52	7

$$D(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y) \tag{4}$$

where $d(x, y)$ is the distance between data objects x and y. The diameter of cluster C_i is defined as

$$\delta(C_i) = \max_{x,y \in C_i} d(x, y) \tag{5}$$

Then, the *Dunn* index can be constructed as

$$Dunn = \min_{i=1,\ldots,K} \left(\min_{\substack{j=1,\ldots,K \\ j \neq i}} \left(\frac{D(C_i, C_j)}{\max_{k=1,\ldots K} \delta(C_k)} \right) \right) \tag{6}$$

where K is the number of clusters. Larger values of *Dunn* Index correspond to better clusters.

III. PSO-BASED AUTOMATIC CLUSTERING

A. The Novel Cluster-recovering Rule

In this part, we describe a new cluster-recovering rule, nearest multiple prototype (NMP) rule, developed recently by the authors. Distance measure is used as similarity measure and the distance of two elements (data object or cluster) a and b is defined as $D(a, b)$. The distance between data object O_i and cluster C_h is given by

$$D(O_i, C_h) = \min\{D(O_i, O_j), D(O_i, m_h) \mid O_j \in C_h\} \tag{7}$$

where m_h is the centroid of cluster C_h. First, every object is allocated to the cluster closest to it and all of the objects allocated to the same cluster make up a set called the candidate objects set. Here, we call this cluster an undetermined cluster of these objects. Then, for each cluster, choose the object nearest to it from its candidate objects set and then gather these objects of all undetermined clusters into a set called the nearest objects set. Finally, in the nearest objects set, the object whose distance to its undetermined cluster is shortest of all is assigned to its undetermined cluster. Repeat the steps above until every object is assigned to cluster. The procedure of our novel rule is described formally as follow.

Let there be n data objects, $O_1, O_2, \ldots, O_n$. K prototypes $P = \{m_1, m_2, \ldots, m_k\}$, $k = 1,2,\ldots,K$, are abstracted from particle p. The distance between every two objects D (Oi, Oj) is calculated before using the rule. A candidate data objects set S_h is relative to C_h. The nearest objects set is denoted as T.

Step 1: Calculate the distance between unassigned object O_i and cluster C_h, $D(O_i, C_h)$, $i = 1,2,\ldots,n$, $h=1,2,\ldots,K$.

Step 2: For each unassigned object O_i, if $D(O_i, C_h) = \min\{ D(O_i, C_b) \mid b = 1,\ldots,K_{max} \}$, add O_i to Set S_h and C_h becomes its undetermined cluster named U_i.

Step 3: For each cluster C_h, $h = 1,2,\ldots,K$, if $D(O_i, C_h) = \min\{ D(O_c, C_h) \mid Oc \in S_h$, then put object O_i of S_h into T.

Step 4: In set T, the object O_i whose distance to the prototype of its undetermined cluster $D(O_i, U_i)$ is shortest of all is assigned to U_i.

Step 5: Repeat steps 2 to step 4 until every object is assigned to cluster.

The nearest prototype (NP) rule adopts one prototype per class and the objects cluster around it. That is the main reason why the algorithms with NP rule usually generate hyperspherical or hyperelliptic clusters. Our novel rule adopts a strategy named multiple prototypes to avoid this tendency. At first, the cluster centers extracted from a genotype become prototypes of each cluster. Then when a new object is assigned to a cluster, it becomes one of the prototype of this cluster for other unsigned objects. With this multiple prototype strategy and modified definition of distance between objects and clusters (7), the process of assignment of data objects is dynamic and flexible. The data points does not rely on only one center to gather around it and the generation of cluster can extend in arbitrary direction from the starting point so that it can discover clusters of not only hyperspherical shape in a more data-driven way rather than impose a specific type of structure (like spherical clusters) to the data.

It's worth noting that in this rule, the cluster centers extracted from a genotype are often not the centers of clusters geometrically but the starting point of the procedure of generation of clusters. Besides, they determine the number of clusters at the beginning of the assignment procedure, which makes the clusters not the hierarchical structure. For these reasons, each of the cluster centers is the special prototype in its cluster and is of great importance in the partitional procedure of our rule.

B. Particle Encoding and Population Initialization

In our experiment, we adopt the centroid-based particle representation scheme proposed by Das et al [16], which can determine automatically the optimal number of clusters while searching for their corresponding partitions. The particle is encoded as a $K_{max} + K_{max} * d$ real numbers vector $Z_i = (T_{i1}, T_{i2}, \ldots, T_{iKmax}, m_{i1}, m_{i2}, \ldots, m_{iKmax})$, where d is the dimension of data for clustering; K_{max} is a user-specified maximum number of clusters; m_{ij} is a cluster center

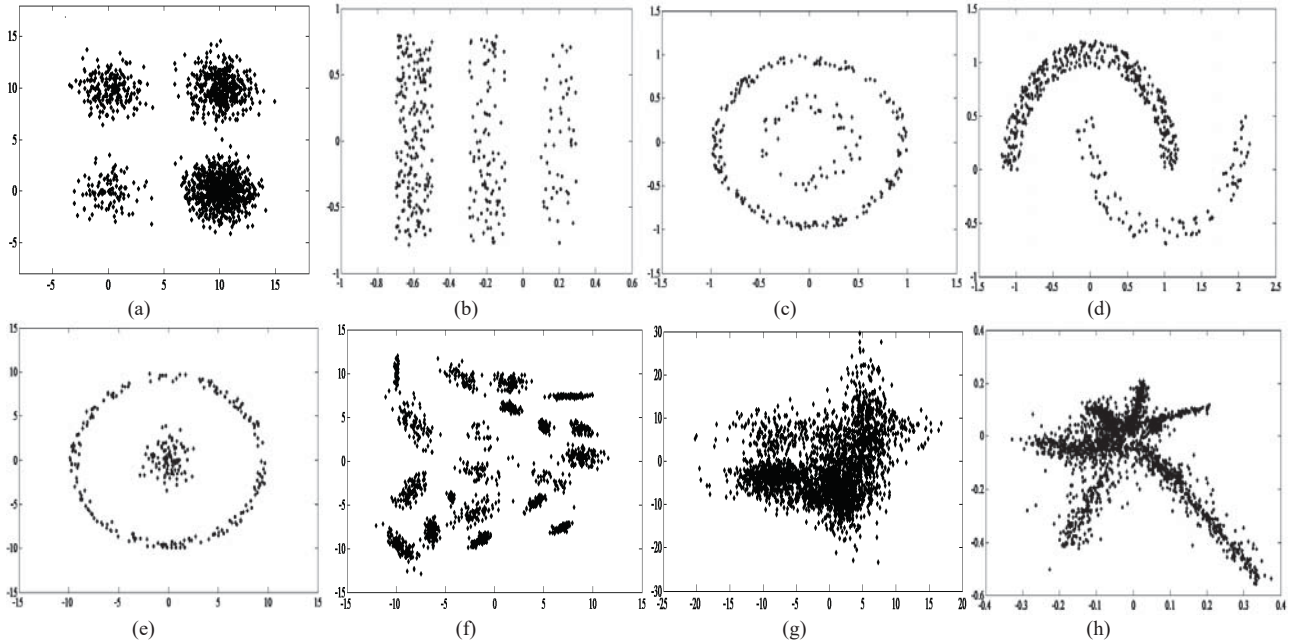

Fig. 1. Synthetic datasets. (a) Four-GussianClusters. (b) Three-rectangles. (c) Two-rings. (d) Half-rings. (e) Ring-GaussianCluster. (f) 2d4c. (g) 10d10c (projected in two dimensions). (h) 100d10c (projected in two dimensions)

coordinate vector; T_{ij}, $j = 1 \ldots K_{max}$, is activation threshold restricted in the interval [0,1] to govern the extraction of the centroids. If T_{ij} is greater than 0.5, the cluster centroid m_{ij} is active (is chosen); otherwise, the cluster centroid is inactive (is not chosen).

For the initial population, in original version [16], the activation thresholds are randomly set within [0, 1]. The cluster centers are also randomly initialized in each dimension in a reasonable range according to the feature of the experimental data. However, the randomly initialization of the activations will lead to the uneven dispersion of the particles from perspective of the number of activated cluster centers. To be specific, obviously the potential number of activated cluster centers, denoted as X, obeys a binomial distribution, the probability of which is

$$P(X=k)=\binom{K_{\max}}{k}(\frac{1}{2})^{K_{\max}}, k=0,1,...,K_{\max} \quad (8)$$

Therefore, the number of particles will decrease from the middle to the sides in distribution, which means the particles with X near ½ K_{max} are much more than that with X near 0 or K_{max}. As a result, the degree of difficulty to search different numbers of clusters for the clustering algorithm is of big difference and the algorithm with this random initialization method will tend to locate the partitions with near ½ K_{max} number clusters. When the actual number far away from ½ K_{max}, this tendentiousness to the number of clusters caused by the initial populations will waste evolution time to move particles to optimal location and it is undesirable for automatic clustering.

Here, we propose a new initialization method to make the tendentiousness vanish and speed up the automatic clustering algorithm. The random initialization of cluster centroids is still adopted as the original version. For the uniform distribution of the particles, the number of activated cluster centers is set properly instead of being randomly initialized. Concretely, in each particle, we can indicate specific number of the activation thresholds, which are randomly chosen, to be randomly set in the range of (0.5, 1] and the left to be randomly set in the range of [0, 0.5] so that the numbers of activated centroids of all particles make up an arithmetic progression. As an example, consider the setting that K_{max} is 5, K_{min} is 2, and the number of particles is 8. The sequence of numbers of activated centroids for 8 particles can be [2 3 4 5 2 3 4 5]. Other permutations of these 8 number are also possible settings. As a result, every value of X, (2, 3, 4, 5), is associated with the same number 2 of particles, which ensures particles uniformly distributed so that the trend to the number of clusters will not exist.

C. Fitness Function

In EAs-based clustering algorithm, the clustering validity indexes are usually used as fitness functions (criterion functions) to evaluate the clustering performance. Each cluster validity index has its bias and come with its own advantages and disadvantages [33], [34]. For evaluating the performance of the proposed algorithm objectively, in our comparative trial, after testing every indexes among the *CH* index [17], the *DB* index [26], the *SIL* index [27], the *Dunn* index [18], we have chosen *CH* index and *Dunn* index which are shown more suitable for our experimental data than others and have been introduced in section II (B).

Both of two indexes are larger when the clustering result is better. Therefore, the object of optimization is to maximize the validity index for achieving proper clustering. The fitness function for particle i is defined as

$$f(P_i)=index(P_i) \quad (9)$$

where P_i is the phenotype of particle.

TABLE II

CLUSTERING QUALITY OF ACPSO ALGORITHMS, DBSCAN AND PSO-BASED CLUSTERING, USING THE DUNN INDEX AS THE FITNESS FUNCTION FOR THE LATTER TWO ALGORITHMS , IN TERMS OF THE RAND AND ADJUST RAND INDEXES. GIVEN ARE THE MEAN AND STANDARD DEVIATION BASED ON 40 RUNS. THE BEST RESULTS IN TERM OF MEAN VALUES FOR EACH DATA SET IS HIGHLIGHTED IN BOLD

Data Set	ACPSO		DBSCAN		PSO-based Clustering	
	R	AR	A	AR	R	AR
Three-rectangles	**1.0000±0.0000**	**1.0000±0.0000**	0.9992±0.0000	0.9984±0.0000	0.8384±0.1625	0.6376±0.3653
Two-rings	**1.0000±0.0000**	**1.0000±0.0000**	0.9941±0.0000	0.9865±0.0000	0.3868±0.0100	0.0606±0.0101
Half-rings	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	**1.0000±0.0000**	0.6630±0.1694	0.3037±0.1922
Ring Gaussian Cluster	**1.0000±0.0000**	**1.0000±0.0000**	0.9740±0.0000	0.9478±0.0000	0.5793±0.0398	0.2004±0.1114

TABLE III

CLUSTERING QUALITY OF ACPSO ALGORITHMS, DBSCAN AND PSO-BASED CLUSTERING, USING THE CH INDEX AS THE FITNESS FUNCTION FOR THE LATTER TWO ALGORITHMS , IN TERMS OF THE RAND AND ADJUST RAND INDEXES. GIVEN ARE THE MEAN AND STANDARD DEVIATION BASED ON 40 RUNS. THE BEST RESULTS IN TERM OF MEAN VALUES FOR EACH DATA SET IS HIGHLIGHTED IN BOLD

Data Set	ACPSO		DBSCAN		PSO-based Clustering	
	R	AR	R	AR	R	AR
2d4c	0.9896±0.0002	0.9756±0.0004	**0.9943±0.0000**	**0.9866±0.0000**	0.9930±0.0000	0.9835±0.0000
2d10c	0.9823±0.0013	0.9125±0.0073	**0.9909±0.0006**	**0.9570±0.0030**	0.9586±0.0096	0.7731±0.0612
2d20c	**0.9975±0.0012**	**0.9758±0.0121**	0.9929±0.0000	0.9311±0.0000	0.9919±0.0223	0.9190±0.0233
10d4c	0.9942±0.0036	0.9858±0.0088	0.9446±0.0007	0.8625±0.0018	**0.9953±0.0084**	**0.9885±0.0195**
10d10c	**0.9550±0.0001**	**0.8352±0.0004**	0.8924±0.0003	0.6216±0.0011	0.8700±0.0882	0.6022±0.0055
10d20c	**0.9998±0.0007**	**0.9979±0.0064**	0.9774±0.0001	0.8133±0.0009	0.9621±0.0661	0.7826±0.1289
50d4c	**0.9557±0.0040**	**0.9113±0.0080**	0.9492±0.0008	0.8985±0.0016	0.7113±0.0024	0.4228±0.0047
50d10c	**0.9943±0.0012**	**0.9736±0.0051**	0.8955±0.0000	0.6019±0.0003	0.8558±0.0189	0.4205±0.0372
100d4c	**0.9585±0.0195**	**0.9058±0.0411**	0.9505±0.0000	0.8878±0.0000	0.7568±0.0030	0.4223±0.0066
100d10c	**0.9808±0.0086**	**0.9155±0.0351**	0.8984±0.0000	0.5645±0.0003	0.7693±0.0175	0.2913±0.0242
Iris	**0.8912±0.0064**	**0.7567±0.0141**	0.8945±0.0034	0.7488±0.0082	0.8737±0.0000	0.7163±0.0000
Wine	**0.8213±0.1043**	**0.6357±0.1895**	0.7268±0.0000	0.4234±0.0000	0.6810±0.0000	0.3702±0.0000
Breast Cancer	0.9214±0.0080	0.8414±0.0613	0.8650±0.0018	0.7301±0.0037	**0.9240±0.0000**	**0.8465±0.0000**
Four-GussianClusters	**1.0000±0.0000**	**1.0000±0.0000**	0.9983±0.0005	0.9963±0.0011	**1.0000±0.0000**	**1.0000±0.0000**

D. Procedure of ACPSO

The procedure of ACPSO algorithm is summarized below:

Step 1: Initialize K_{max} activation thresholds of each particle in range [0, 1] in a specific way that is described in detail in Part B. to make sure that particles obey uniform distribution from point of the number of activated cluster centers. K_{max} cluster centroids are randomly initialized.

Step 2: Initialize the velocity of each particle randomly in around 10%–20% of the dynamic range of the variable on each dimension.

Step 3: For each particle in the population

1) The clusters centers encoded in it are extracted.

2) Perform clustering by assigning each object to the particular cluster center by means of the NMP rule.

3) Compute fitness.

Step 4: Update the global best and local best positions. Then update the velocities and positions of particles using (1) and (2).

Step 5: Loop to step 3 until the maximum number of iterations is reached by when the cluster partition obtained from the global best is the final solution.

IV. EMPIRICAL RESULTS AND DISCUSSION

In this section, we compare performance of the ACPSO algorithm with the density-based clustering algorithm, DBSCAN [28] and the PSO-based clustering algorithm using the NP rule, which uses the same particle representation scheme and fitness function as the ACPSO. Moreover, the efficiency of the new population initialization method was validated.

A. Experimental Settings

Fifteen synthetic and three real data sets were used in our experiment, summarized in Table I and shown in Fig. 1. The synthetic data sets used in this study are handcrafted (Four-GussianClusters, Three-rectangles, Two-rings, Half-rings and Ring Gaussian Cluster) or randomly generated (2d4c, 2d10c, 2d20c, 10d4c, 10d10c,10d20c, 50d4c, 5010c, 100d4c and 100d10c) from two cluster generators invented by Handl and Knowles [29], downloaded from the website http://pers-onalpage.manchester.ac.uk/mbs/Julia.Handl/gener-ators.html. The three real data sets (Iris, Wine, Breast Cancer) can be downloaded from the UCI Machine Learning Repository at http://archive.ics.uci.edu/ml/index.-html [30].

The parameters settings for ACPSO and PSO-based clustering algorithm using NP rule are the same. The swarm size is 40 and each particle has two neighbors. The inertia weight w is set 0.75. The legal velocity range Vmax is set to 20% of the search range. The c_1 and c_2 are both assigned 2 recommended by [21]. The minimum and maximum number of clusters are set 2 and 30 respectively for all datasets. When comparing two initialization methods, the range of

TABLE IV
COMPARISON OF CLUSTERING QUALITY OF DBSCAN AGAINST PSO-BASED CLUSTERING AND ACPSO ALGORITHMS WITH WILCOXON RAN SUM TEST. THE ANALYSIS IS BASED ON THE RESULTS IN TABLE III AND TABLE IV USING THE ADJUSTED RAND INDEX

Data Set	ACPSO vs			
	DBSCAN		PSO-based Clustering	
	p-value	Signi-ficance	p-value	Signi-ficance
Four-Gussian Clusters	3.8683×10^{-17}	ES	–	–
Three-rectangles	6.5292×10^{-17}	ES	4.2899×10^{-7}	ES
Two-rings	6.5292×10^{-19}	ES	1.9649×10^{-16}	ES
Half-rings	–	–	9.5412×10^{-17}	ES
Ring-Gaussian Cluster	6.5292×10^{-19}	ES	1.9353×10^{-16}	ES
2d4c	5.1824×10^{-18}	ES	5.1824×10^{-18}	ES
2d10c	1.1441×10^{-14}	ES	1.4050×10^{-14}	ES
2d20c	1.9514×10^{-16}	ES	4.7003×10^{-14}	ES
10d4c	1.5283×10^{-15}	ES	0.1172	NS
10d10c	7.3561×10^{-16}	ES	2.2095×10^{-15}	ES
10d20c	1.5462×10^{-16}	ES	7.0345×10^{-16}	ES
50d4c	3.5435×10^{-10}	ES	1.4638×10^{-15}	ES
50d10c	6.3577×10^{-17}	ES	2.8028×10^{-16}	ES
100d4c	3.9082×10^{-11}	ES	1.1331×10^{-14}	ES
100d10c	1.0596×10^{-14}	ES	8.4181×10^{-15}	ES
Iris	4.4423×10^{-08}	ES	3.5286×10^{-17}	ES
Wine	0.0135	S	1.5863×10^{-16}	ES
Breast Cancer	1.4035×10^{-15}	ES	0.0394	S

NS: No Significant S: Significant ES: Extremely Significant

number of clusters for each dataset is set as the Table VII shows. The terminal condition is 2000 iteration times. Table I summarizes the settings for the DBSCAN, we choose an optimal set of parameters (neighborhood's radius ε and a minimum number of neighborhood's objects MinPts) for each dataset after testing many possibilities settings in term of *CH* index or *Dunn* index.

The clustering quality is judged by external criteria in our experiment, which can be regarded as objective evaluation [1]. The Rand (R) Index [31] and the Adjusted Rand (AR) Index [32] are chosen. Large value of R or AR Index means close agreement between the two partitions.

All the results are based on 40 dependent runs of the clustering algorithm and have been reported in terms of the averages and standard deviations over all simulations. The best results in term of mean value are shown in bold.

B. Experimental Results and Discussions

The clustering quality of three clustering algorithms by *Dunn* index are illustrated in Table II and by *CH* index is illustrated in Table III. Besides, the two-sided and non-parametric Wilcoxon rank sum tests are conducted, as summarized in Table IV. Table V and Table VI summarizes the number of clusters found by three algorithms with *Dunn* index and *CH* index respectively.

TABLE V
NUMBER OF CLUSTERS FOUND BY ACPSO ALGORITHMS, DBSCAN AND PSO-BASED CLUSTERING, USING THE DUNN INDEX AS THE FITNESS FUNCTION FOR THE LATTER TWO ALGORITHMS. GIVEN ARE THE MEAN AND STANDARD DEVIATION BASED ON 40 RUNS.

Data Set	Actual Number of Clusters	ACPSO	DBSCAN	PSO-based Clustering
Three-rectangles	3	**3.0000 ±0.0000**	**3.0000 ±0.0000**	10.2000 ±8.3102
Two-rings	2	**2.0000 ±0.0000**	**2.0000 ±0.0000**	19.1250 ±3.6687
Half-rings	2	**2.0000 ±0.0000**	**2.0000 ±0.0000**	6.7750 ±6.5821
Ring Gaussian Cluster	2	**2.0000 ±0.0000**	**2.0000 ±0.0000**	10.875 ±5.05068

On handcrafted datasets, as Table II, III, V and VI show, since DBSCAN is well known for its ability to discover clusters of arbitrary shape, it is not surprising to observe that it can find exactly correct number of clusters with high clustering quality. A strong performance of the PSO-based clustering algorithm on the Four-GussianClusters can be expected, because NP rule usually generate hyperspherical or hyperelliptic clusters as we have analyzed in III. A. Therefore, it also can be expected that its performance breaks down drastically on other handcrafted datasets with nonconvex shaped clusters, especially on dataset Two-rings. In contrast, the ACPSO has found out the true partitions without any misclassification. The advantage of the ACPSO that it can well partition datasets of not only hyperspherical- or hyperelliptic-shaped clusters is supported by these experimental results and comparisons.

On randomly generated datasets, they are all elliptic, elongated and have different number of dimensions and clusters with different degrees of overlap. Therefore, we focus on comparing the algorithms in terms of the ability in partitioning convex, elongated-shaped and overlapped clusters and the scalability when the dimensions and number of clusters increase. As the Tables II and IV clearly indicate, the proposed ACPSO performs extremely significantly better than DBSCAN and PSO-based clustering algorithm using NP rule on most randomly generated data sets except 2d4c, 2d10c, 10d4c in terms of clustering quality, at a 5% significance level.

On real data sets, used for test robustness of our algorithm to noise and overlap of data, Table III reveals that the ACPSO outperforms other two algorithms on dataset Iris and Table IV testifies that the differences are extremely significant. Table VI indicates that all algorithms yield three clusters of dataset Iris on each run even though the two of three clusters are considerably overlapping. On dataset Wine, it can be observed from Table III and VI that the ACPSO also stands out in terms of both clustering quality and number of clusters found. ACPSO and PSO-based clustering algorithm achieve nearly the same value of AR index on dataset Breast Cancer. DBSCAN performs worse on this dataset and all the differences are statistically significant as Table IV shown.

TABLE VI
NUMBER OF CLUSTERS FOUND BY ACPSO ALGORITHMS, DBSCAN AND PSO-BASED CLUSTERING, USING THE CH INDEX AS THE FITNESS FUNCTION FOR THE LATTER TWO ALGORITHMS. GIVEN ARE THE MEAN AND STANDARD DEVIATION BASED ON 40 RUNS.

Data Set	Actual Number of Clusters	ACPSO	DBSCAN	PSO-based Clustering
2d4c	4	**4.0000 ±0.0000**	3.0000 ±0.0000	**4.0000 ±0.0000**
2d10c	10	11.4500 ±0.5454	**9.0000 ±0.0000**	16.8500 ±1.7965
2d20c	20	**20.9750 ±0.7902**	19.0000 ±0.0000	21.9250 ±1.9543
10d4c	4	**4.0000 ±0.0000**	3.0000 ±0.0000	3.9750 ±0.1561
10d10c	10	6.0000 ±0.0000	**7.0000 ±0.0000**	4.0250 ±0.3527
10d20c	20	**19.9000 ±0.3000**	16.0000 ±0.0000	13.7000 ±2.5219
50d4c	4	**4.0000 ±0.0000**	3.0000 ±0.0000	5.0000 ±0.0000
50d10c	10	8.9750 ±0.15612	6.0000 ±0.0000	**10.0250 ±1.1289**
100d4c	4	5.5500 ±0.7730	**4.0000 ±0.0000**	5.0000 ±0.0000
100d10c	10	**8.5500 ±0.5895**	13.0000 ±0.0000	7.3250 ±0.4684
Iris	3	**3.0000 ±0.0000**	**3.0000 ±0.0000**	**3.0000 ±0.0000**
Wine	3	**2.6000 ±0.4899**	2.0000 ±0.0000	2.0000 ±0.0000
Breast Cancer	2	**2.0000± 0.0000**	**2.0000 ±0.0000**	**2.0000 ±0.0000**
Four-Gussian Clusters	4	**4.0000 ±0.0000**	**4.0000 ±0.0000**	**4.0000 ±0.0000**

To investigate the effect of the new population initialization method, we have compare the ACPSO using the original and the new population initialization methods. We run two versions of ACPSO algorithms on some of datasets and stop as soon as the algorithm find the proper number of cluster, as well as the preset threshold fitness value. The setting of minimum and maximum number of clusters and results on eight randomly generated data sets are summarized in Table VII. From the results, we can observe that the ACPSO with the proposed initialization method was able to reach the cutoff fitness value and cluster number within fewer number of iterations than the original method on all test datasets.

V. CONCLUSION

In this paper, in the field of partitional clustering algorithms based on EAs, we presented a novel clustering approach using PSO, named the automatic clustering PSO (ACPSO) algorithm. It was able to automatically find an appropriate number of clusters in the procedure of clustering and partition datasets with not only convex but also nonconvex shaped clusters. A new decoding method named the nearest multiple prototype (NMP) rule is used, which is designed by applying multiple prototypes strategy in traditional nearest prototypes (NP) rule. Besides, we improve the particle initialization method proposed by Das et al to avoid its tendentiousness of locating the partitions with near ½ K_{max} number clusters. The experimental results demonstrate that the proposed ACPSO is able to outperform the DBSCAN and PSO-based clustering algorithm with the traditional NP rule in a statistically meaningful way over a majority of the benchmark data sets tested contributing to higher solution accuracy, more accurate estimated number of clusters. The good scalability of ACPSO is also be demonstrated by the experimental results. Besides, we can observe that the new initialization method exactly accelerates the proposed clustering algorithm.

TABLE VII
MEAN AND STANDARD DEVIATION OF THE NUMBER OF ITERATIONS REQUIRED AND NUMBER OF FAILURE OVER 40 RUNS FOR ACPSO AND PSO-BASED CLUSTERING ALGORITHMS TO REACH A PREDEFINED CUTOFF FITNESS VALUE AND A CUT OFF RANGE OF NUMBER OF CLUSTERS. GIVEN ARE THE MEAN AND STANDARD DEVIATION BASED ON 40 RUNS. THE BEST RESULTS IN TERM OF MEAN VALUES FOR EACH DATA SET IS HIGHLIGHTED IN BOLD

Data Set	K_{min}	K_{max}	Cut off Fitness Value	Cut off Cluster Number	ACPSO	PSO-based
2d4c	2	30	4800	4	**74.1740 ±95.0352**	123.7500 ±69.2090
2d10c	2	60	10000	8-12	**408.9250 ±362.0265**	1742.2250 ±452.9313
10d4c	2	30	490	4	**15.8000 ±8.4770**	50.5250 ±19.82043
10d10c	2	60	580	6-14	**150.2500 ±108.1059**	534.7000 ±348.0321
50d4c	2	30	480	4	**119.5500 ±136.2571**	180.1500 ±240.7958
50d10c	2	60	650	8-12	**657.7250 ±528.8392**	1077.9000 ±757.5246
100d4c	2	30	380	3-6	**277.6750 ±379.5610**	406.1750 ±501.2074
100d10c	2	60	550	8-12	**824.1250 ±591.3673**	1279.7500 ±778.3124

REFERENCES

[1] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1998.

[2] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Nat.Acad. Sci. USA*, vol. 95, no. 25, pp. 14 863–14 868, Dec. 1998.

[3] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data—An Introduction to Cluster Analysis*. Series in Probability and Mathematical Statistics. New York: Wiley, 1990.

[4] E. Falkenauer, *Genetic Algorithms and Grouping Problems*. Chichester, U.K.: Wiley, 1998.

[5] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, no. 3, pp. 768–780, 1965.

[6] D. Steinley, "K-means clustering: A half-century synthesis," *Brit. J. Math. Stat. Psychol.*, vol. 59, pp. 1–34, May 2006.

[7] D. Fogel, "An introduction to simulated evolutionary optimization," *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 3–14, Jan. 1994. F. Glover, "Tabu search, part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, 1989.

[8] W. N. Chen, J. Zhang, H.S.H. Chung, W. L. Zhong, W. G. Wu and Y. H. Shi, "A novel set-based particle swarm optimization method for

discrete optimization problem," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278-300, 2010.

[9] F. Glover, "Tabu search: Part II," *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, 1990.

[10] G. Phanendra Babu and M. Narasima Murty, "Clustering with evolution strategies," *Pattern Recognit.*, vol. 27, no. 2, pp. 321–329, 1994.

[11] U. Maulik and I. Saha, "Automatic fuzzy clustering using modified differential evolution for image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 9, pp. 3503–3510, Sep. 2010.

[12] K. S. Al-Sultan, "A tabu search approach to the clustering problem," *Pattern Recognit*., vol. 28, no. 9, pp. 1443–1451, 1995.

[13] P. C. H. Ma, K. C. C. Chan, X. Yao, and D. K. Y. Chiu, "An evolutionary clustering algorithm for gene expression microarray data analysis," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 296–314, Jun. 2006.

[14] E. Hruschka, R. Campello, A. Freitas, and A. Carvalho, "A survey of evolutionary algorithms for clustering," *IEEE Trans. Syst., Man, Cybern.C*, *Appl. Rev.*, vol. 39, no. 2, pp. 133–155, Mar. 2009.

[15] L. I. Kuncheva and J. C. Bezdek, "Selection of cluster prototypes from data by a genetic algorithm," in *Proc. 5th Eur. Congr. Intell. Tech. Soft Comput.*, 1997, pp. 1683–1688.

[16] S. Das, A. Abraham, and A. Konar, "Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm," *Pattern Recognit. Lett.*, vol. 29, no. 5, pp. 688–699, Apr. 2008.

[17] R. Cali´nski and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Stat.*, vol. 3, no. 1, pp. 1–27, 1974.

[18] J. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters," *J. Cybern.*, vol. 3, no. 3, pp. 32–57, 1974.

[19] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micromachine Hum. Sci.*, 1995, pp. 39–43.

[20] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Nov.–Dec. 1995, pp. 1942–1948.

[21] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence. San Diego*, CA: Academic, 2001.

[22] W. N. Chen, J. Zhang, Y. Lin, N. Chen, Z. H. Zhan, H. Chung, Y. Li, and Y. H. Shi, "Particle swarm optimization with an aging leader and challengers," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 241–258, Apr. 2013.

[23] Q. Yang, W. N. Chen, T. Gu, H. Zhang, J. D. Deng, Y. Li and J. Zhang. "Segment-Based Predominant Learning Swarm Optimizer for Large-Scale Optimization," *IEEE Trans. Cybern.*, in press, 2016

[24] M. Shen, Z. Zhan, W. Chen, Y. J. Gong, J. Zhang, Y. Li, "Bi-Velocity Discrete Particle Swarm Optimization and Its Application to Multicast Routing Problem in Communication Networks," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, Dec. 2014

[25] Y. J. Gong, J. Zhang, H. S. H. Chung, W. N. Chen, Z. H. Zhan, Y. Li, and Y. H. Shi, "An Efficient Resource Allocation Scheme Using Particle Swarm Optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 6, Dec. 2012

[26] D. Davies and D. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.

[27] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis.* New York: Wiley, 1990.

[28] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD'96)*, 1996, pp. 226–231.

[29] J. Handl and J. Knowles, "Improving the scalability of multiobjective clustering," in *Proc. Congr. Evol. Comput.*, 2005, vol. 3, pp. 2372–2379.

[30] A. Asuncion and J. Newman, *UCI Machine Learning Repository*, Irvine, CA, School Inf. Comput. Sci., Univ. California, 2007. [Online].Available:http://www.ics.uci.edu/~mlearn/MLRepository.html

[31] W. Rand, "Objective criteria for the evaluation of clustering methods," *J. Amer. Statist. Assoc.*, vol. 66, no. 336, pp. 846–850, 1971.

[32] A. Hubert, "Comparing partitions," *J. Classification*, vol. 2, pp. 193–198, 1985.

[33] J. Xu, D. C. Wunsch, "A comparison study of validity indices on swarm-intelligence-based clustering", *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42 (2012) 1243-1256.

[34] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part I*," ACM SIGMOD Rec.*, vol. 31, no. 2, pp. 40–45, Jun. 2002.