Computing and Informatics, Vol. 36, 2017, 637–663, doi: 10.4149/cai_2017_3_637

A NEW APPROACH OF DYNAMIC CLUSTERING BASED ON PARTICLE SWARM OPTIMIZATION AND APPLICATION IN IMAGE SEGMENTATION

Dang Cong Tran

State Key Lab of Software Engineering
School of Computer, Wuhan University
Wuhan 430072, China
&
Vietnam Academy of Science and Technology
Hanoi, Vietnam
e-mail: trandangcong@gmail.com

Zhijian Wu

State Key Lab of Software Engineering
School of Computer, Wuhan University
Wuhan 430072, China
e-mail: zhijianwu@whu.edu.cn

Abstract. This paper presents a new approach of dynamic clustering based on improved Particle Swarm Optimization (PSO) and which is applied to image segmentation (called DCPSONS). Firstly, the original PSO algorithm is improved by using diversity mechanism and neighborhood search strategy. The improved PSO is then combined with the well-known data clustering k-means algorithm for dynamic clustering problem where the number of clusters has not yet been known. Finally, DCPSONS is applied to image segmentation problem, in which the number of clusters is automatically determined. Experimental results in using sixteen benchmark data sets and several images of synthetic and natural benchmark data demonstrate that the proposed DCPSONS algorithm substantially outperforms other competitive algorithms in terms of accuracy and convergence rate.

Keywords: Particle swarm optimization, neighborhood search, diversity, global optimization, dynamic clustering, image segmentation

Mathematics Subject Classification 2010: 68T01

1 INTRODUCTION

Data clustering is a process of identifying natural groupings or clusters, within multidimensional data, based on some similarity measures (e.g. Euclidean distance). Due to raising amount of data from wide variety fields, clustering has become an increasingly important topic in recent years. Clustering is a method of unsupervised learning where the whole objective is to group a set of elements into a cluster and the elements within that cluster have a high degree of similarity, whereas elements belonging to different clusters have a high degree of dissimilarity. Hierarchical and partitioning methods are the most popular clustering techniques. Hierarchical clustering finds a sequence of partitions where the algorithm starts from one group with all objects and is executed until it finds singletons groups, or vice versa, whereas partitioning clustering directly divides data objects into some fixed number of clusters [1, 2, 3, 4, 5, 6] using a suitable objective function. An advantage of partitioning method is its ability to manipulate large data sets. When the number of clusters is known a priori clustering may be formulated as distribution of objects in multi-dimensional space among groups in such a way that objects in the same cluster are more similar in some sense than those in different clusters. This involves minimization of some extrinsic optimization criterion.

The k-means clustering algorithm was developed by Hartigan [1] which is one of the most popular and widely used clustering techniques because it is easy to implement and efficient, with linear time complexity. However, k-means algorithm can only discover spherical clusters and it is sensitive to the selection of initial points, which causes that it may converge to local optimum solution.

In the last decade, various clustering methods based on intelligent optimization algorithms have been introduced to overcome the above mentioned problems [6, 7, 8, 9]. Particle swarm optimization (PSO) was first proposed by Kennedy et al. [10], and it is one of swarm intelligent optimization algorithms. This algorithm simulates bird flocking or fish schooling behavior to achieve a self-evolution system. It can search automatically the optimum solution in the vector space. However, there exists a drawback of premature convergence in the PSO. To improve the PSO algorithm over the drawback, many literatures presented its approaches, some of them employed method of neighborhood search and diversity mechanism [11, 12].

An alternative approach is to apply evolutionary algorithms (EAs) to clustering, yielding EA-based clustering algorithms. Unlike k-means clustering, they simultaneously optimize a population of candidate solutions, which gives them the ability to escape from local optima. Various EA-based clustering algorithms have been developed, including genetic algorithm [13], differential evolution [14, 15], ant colony optimization [16], artificial bee colony [17], and particle swarm optimization [18, 19, 20, 21]. In details, Neshat et al. [7] proposed a method called PSOK, in which improved PSO algorithm and k-means algorithm were combined. A method called CPSOII was proposed by Zhang et al. [8], in this approach the authors used the dynamic crossover to enhance the PSO algorithm. In [22], Jiang et al. proposed an approach called PSOAG, in which the authors proposed a concept of age to measure the search ability of each particle in local area.

Most clustering algorithms, like k-means, require the number of clusters to be specified in advance. Finding the optimum number of clusters in a data set is usually a challenge since it requires a priori knowledge, and/or ground truth about the data, which is not always available. The problem of finding the optimum number of clusters in a data set has been the subject of several research efforts [9, 22, 23, 24], however, despite the amount of research in this area, the outcome is still unsatisfactory [25]. An alternative solution for this problem is to use dynamic clustering techniques. Dynamic clustering techniques have two general objectives, finding the optimal number of clusters and partitioning the data objects into clusters. Several approaches based on EAs such as PSO, ABC, ACO were presented, namely, Omran et al. [9] proposed an approach called DCPSO, in which the binary PSO was employed for the first phase of finding the number of clusters, then k-means was applied to refine the chosen clusters. To improve the idea of DCPSO, Kuo et al. [26] presented the DCPG approach, in which the hybrid of PSO and GA was used. Further, Masoud et al. [27] presented a new approach CPSOII which automatically finds the best number of clusters and simultaneously categorizes data objects.

The proposed approach in this paper is called DCPSONS, in the approach we improve the idea presented in [9, 26] which first uses the binary PSO to find the number of clusters, k-means is then applied to refine the centers of the chosen clusters. Further, with the aim to improve the performance of PSO, the proposed DCPSONS approach employs the neighborhood search strategy and diversity mechanism, and combines with the k-means clustering technique. Thus, DCPSONS can automatically determine the optimal number of clusters and cluster the data set with minimal user interference.

Image segmentation is a fundamental component in many computer vision applications [5]. Image segmentation is defined as the process of subdividing an image into its constituent parts and extracting desired parts. There are many methods for image segmentation in the literature [28, 29], of which one of the popular method is to use a clustering algorithm (such as k-means). In this paper, taking the advantage of the clustering algorithm based on EAs, the proposed DCPSONS algorithm is applied to image segmentation where several synthetic and natural images were used to evaluate the algorithm performance.

The rest of the paper is organized as follows: Section 2 briefly reviews the preliminaries and related works of the proposed approaches. The details of proposed DCPSONS approach will be described in Section 3. The experimental results of dynamic clustering will be presented and analyzed in Section 4. The results of application in image segmentation will be demonstrated in Section 5. Finally, in Section 6 the conclusion will be drawn.

2 RELATED WORKS

In partitioning clustering problems, we need to divide a set of N objects into K clusters. Let $O(o_1, o_2, \ldots, o_N)$ be the set of N objects of data set. Each object has D features, and each feature is quantified with a real-value. Let $S_{N \times D}$ be the feature data matrix with N rows and D columns. Each row S_i presents a data vector and s_{ij} corresponding the j^{th} feature of i^{th} data vector ($i = 1, 2, \ldots, N$, $j =, 1, 2, \ldots, D$). Let $C = (C_1, C_2, \ldots, C_K)$ be the K clusters. Then $C_i \neq \phi$, $C_j \cap C_i \neq \phi$, $\cup_{j=1}^{K} C_i = \mathrm{O}$, $i, j = 1, 2, \ldots, K$, $i \neq j$. The goal of clustering algorithm is to find such a C which makes the objects in the same clusters are as similar as possible while other objects in the different clusters as dissimilar, which can be measured by some criterions.

The brief reviews of k-means, PSO, and related techniques of neighborhood search strategy and diversity mechanism are described in the following sections.

2.1 K-Means Clustering

One of the most important components of a clustering algorithm is the measure of similarity used to determine how close two patterns are to each other. K-means clustering [1] groups data vectors into a pre-specified number of clusters, based on Euclidean distance as similarity measure. Euclidean distances among data vectors are small for data vectors within a cluster as compared with distances to other data vectors in different clusters. Vectors of the same cluster are associated with one centroid vector, which represents the "midpoint" of that cluster and is the mean of the data vectors that belong together. The classical k-means algorithm is summarized as follows:

Step 1. Randomly choose K cluster centroids from N objects.

Step 2. For each data vector, assign the vector to the cluster with the closest centroid, where the distance to the centroid is determined by Equation (1).

$$d(S_i, Z_j) = \sqrt{\sum_{p=1}^{D} (s_{ip} - z_{jp})^2}. \tag{1}$$

Step 3. Recalculate the cluster centroids, using Equation (2) as follows:

$$Z_j = \frac{1}{N_{C_j}} \sum_{\forall S_p \in C_j} S_p \tag{2}$$

where N_{C_j} is the number of data vectors in cluster j and C_j is the subset of data vectors that form cluster j, if stopping criterion is not satisfied then return Step 2.

The k-means clustering procedure terminates when any of the following criteria is satisfied: when the maximum number of iterations has been exceeded, when there is a small change in the centroid vectors over a number of iterations, or when there are no cluster membership changes. For the purpose of this research, the algorithm terminates when a user-specified number of iterations has been exceeded.

2.2 Particle Swarm Optimization

Similar to other evolutionary algorithms, PSO [10, 31] is also a population based search algorithm and starts with an initial population of randomly generated solutions called particles. Each particle in PSO has a velocity and a position vector. PSO remembers both the best position found by all particles and the best positions found by each particle in the search process. For a search problem in D-dimensional space, a particle represents a potential solution. The velocity v_{ij} and position x_{ij} of the j^{th} dimension of the i^{th} particle are updated according to Equations (3) and (4) as follows:

$$\begin{aligned} v_{ij}(t+1) = w \cdot v_{ij}(t) &+ c_1 \cdot \mathit{rand1}_{ij} \cdot (\mathit{pbest}_{ij}(t) - x_{ij}(t)) \\ &+ c_2 \cdot \mathit{rand2}_{ij} \cdot (\mathit{gbest}_j(t) - x_{ij}(t)), \end{aligned} \tag{3}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{4}$$

where $i = 1, 2, \ldots, NP$ is the particle's index, NP is the population size, $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ is the position of the i^{th} particle, $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$ represents the velocity of i^{th} particle, $\mathit{pbest}_i = (\mathit{pbest}_{i1}, \mathit{pbest}_{i2}, \ldots, \mathit{pbest}_{iD})$ is the best previous position yielding the best fitness value for the i^{th} particle; and $\mathit{gbest} = (\mathit{gbest}_1, \mathit{gbest}_2, \ldots, \mathit{gbest}_D)$ is the global best particle found by all particles so far, $\mathit{rand1}_{ij}$ and $\mathit{rand2}_{ij}$ are two random numbers independently generated within the range of $[0, 1]$, c_1 and c_2 are two learning factors which control the influence of the social and cognitive components, w is the inertia factor, and $t = 1, 2, \ldots$ indicates the iterations.

2.3 Particle Swarm Optimization with Neighborhood Search

Implementing with neighborhood, various types of connected graphs or topologies may be used, like Ring, Four clusters, Pyramid, Square, etc. Mendes et al. [32] designed four different population topologies, including circle, wheel, star, and random. For PSO, the ring topology is simple and easy to implement.

A k-neighborhood radius in the ring topology, consisting of vector $(X_{i-k}, \ldots, X_i, \ldots, X_{i+k})$, for i^{th} particle, where k is an integer within $\{1, 2, \ldots, \frac{NP-1}{2}\}$, as the neighborhood size must be smaller than the population size $2 \cdot k + 1 \leq NP$. Figure 1 illustrates the k-neighborhood radius, where $k = 2$.

By employment of local neighborhood search and global neighborhood search strategies, H. Wang et al. [11] proposed DNSPSO approach to enhance PSO algorithm. In DNSPSO, for each particle, its neighborhood may cover better solutions. To improve the ability of exploitation, a local neighborhood search (LNS) strategy

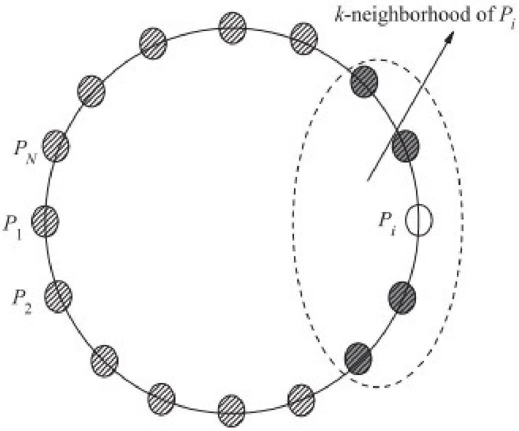


Figure 1: The k-neighborhood in a ring topology, where $k = 2$

is proposed. During searching the neighborhood of a particle P_i, a trial particle $L_i = (LX_i, LV_i)$ is generated by Equations (5), (6) as follows:

$$LX_i = r_1 \cdot X_i + r_2 \cdot pbest_i + r_3 \cdot (X_c - X_d), \tag{5}$$

$$LV_i = V_i \tag{6}$$

where X_i is the position vector of the i^{th} particle, $pbest_i$ is the previous best particle of P_i, X_c and X_d are the position vectors of two random particles in the k-neighborhood radius of P_i, $c, d \in [i-k, i+k] \wedge c \neq d \neq i$, r_1, r_2 and r_3 are three uniform random numbers within (0,1), and $r_1 + r_2 + r_3 = 1$. The random numbers r_1, r_2 and r_3 are the same for all $j = 1, 2, \ldots, D$, and they are generated anew in each generation. The $pbest_i$ is the previous best particle of X_i, so it is not on the circle topology. To keep the flying direction of P_i, the trial particle L_i keeps the same velocity of P_i.

Besides the LNS, a global neighborhood search (GNS) strategy is proposed to enhance the ability of exploration. When searching the neighborhood of a particle P_i, another trial particle $G_i = (GX_i, GV_i)$ is generated by Equations (7), (8) as follows:

$$GX_i = r_4 \cdot X_i + r_5 \cdot gbest + r_6 \cdot (X_e - X_f), \tag{7}$$

$$GV_i = V_i \tag{8}$$

where $gbest$ is the global best particle, X_e and X_f are the position vectors of two random particles chosen for the entire swarm, $e, f \in [1, NP] \wedge e \neq f \neq i$, r_4, r_5 and r_6 are three uniform random numbers within (0, 1), and $r_4 + r_5 + r_6 = 1$. The random numbers r_4, r_5 and r_6 are the same for all $j = 1, 2, \ldots, D$, and they are generated anew in each generation.

From the idea of DNSPSO, in our previous research [12] we proposed an approach EPSODNS, in which the local search strategy is modified by Equation (9) as follows:

$$LX_i = r_1 \cdot X_i + r_2 \cdot (pbest_i - X_i) + r_3 \cdot nbest_i \tag{9}$$

where $pbest_i$ is the best previous particle of X_i, $nbest_i$ is the best particle of X_i neighborhood. r_1, r_2 and r_3 are three mutually different uniform random numbers within $(0, 1)$, and $r_1+r_2+r_3 = 1$. In EPSODNS, the improvement is the employment of local best particle $nbest_i$ in the updating position equation. By using local best particle, the exploitation ability of search strategy will be improved. The results of EPSODNS on several benchmark functions show that the modified local search strategy is effective and robust in solving numerical global optimization problem.

2.4 Diversity Mechanism

In PSO algorithm, many literatures express that the diversity is very important in searching process. In 2002, Reget et al. [33] presented a diversity-guided particle optimizer, called ARPSO, purposes the attractive and repulsive PSO (ARPSO) in trying to overcome the problem of premature convergence. It uses a diversity measure to control the swarm. The basic PSO algorithm only employs an attraction phase, in which particles are attracted by *pbest* and *gbest*. All particles in the swarm move quickly to the same direction and similarities among particles increase very fast, so the diversity of swarm drops below a lower bound, ARPSO switch to the repulsion phase.

Like DNSPSO [11] and EPSODNS [12] approaches, the diversity mechanism was employed. In these approaches, for each particle $P_i(t)$ a new particle $P_i(t + 1)$ is generated by the PSO's velocity and position updating equations. By recombining $P_i(t)$ and $P_i(t + 1)$, a trial particle $TP_i(t+1) = (TX_i(t+1), TV_i(t+1))$ is generated by Equations (10), (11) as follows:

$$TX_{ij}(t+1) = \begin{cases} X_{ij}(t+1) & \text{if } rand_j(0,1) < P_r, \\ X_{ij}(t) & \text{otherwise,} \end{cases} \tag{10}$$

$$TV_{ij}(t+1) = V_{ij}(t+1) \tag{11}$$

where P_r is a user-defined value of greedy selection probability. After recombination, a greedy selection is used as follows:

$$P_i(t+1) = \begin{cases} TP_i(t+1) & \text{if } f(TP_i(t+1)) < f(P_i(t+1)), \\ P_i(t+1) & \text{otherwise.} \end{cases} \tag{12}$$

In PSO, particles tend to move the same position during the search process. It means that particles become similar with increasing of iterations. When the trial particle $TP_i(t + 1)$ is selected into the next generation, the dissimilarities between

$TP_i(t+1)$ and $P_i(t+1)$ will determine the dissimilarities among $P_i(t+1)$ and the rest particles of the swarm.

3 PROPOSED METHOD

This section will describe the proposed DCPSONS approach for dynamic clustering. The proposed DCPSONS approach in this paper is referred to the DCPSO [9] and DCPG [26], where the binary PSO algorithm presented by Kennedy et al. [38] was first employed to find the optimal number of clusters, k-means was then applied to find the best clustering result. In binary PSO, the position value x_i is restricted to set $\{0,1\}$. The velocity v_i is as a probability to change a bit from 0 to 1, or from 1 to 0 when updating the position of particles. This can be done by using a sigmoid function defined as follows:

$$sigmoid(x) = \frac{1}{1+e^{-x}}. \tag{13}$$

Hence, the equation for updating positions of Equation (4) is replaced by the probabilistic update equation [39] as follows:

$$x_{ij} = \begin{cases} 0 & \text{if } rand_j \geq sigmoid\left(v_{ij}\right), \\ 1 & \text{if } rand_j \prec sigmoid\left(v_{ij}\right) \end{cases} \tag{14}$$

where $rand_j \sim U(0,1)$. Similar to DCPSO [9], DCPSONS also makes use of a binary PSO to optimize the number of clusters. But, in order to improve the algorithm performance in terms of the accuracy and pull out from local optima, in DCPSONS the neighborhood search strategy and diversity mechanism are employed with binary PSO.

The algorithm works as follows: a pool of cluster centroids, M, is randomly chosen from data set S. The swarm of particles, P, is then randomly initialized according to Equation (15) as follows:

$$x_{ij} = \begin{cases} 0 & \text{if } rand_j \geq P_{ini}, \\ 1 & \text{if } rand_j \prec P_{ini} \end{cases} \tag{15}$$

where P_{ini} is a user-specified probability defined by [40] which is used to initialize a particle position. The length of particle is equal to number of cluster centroids, $N_c = |M|$. If the position of particle, x_{ij}, is "1" that means the corresponding point is a centroid; otherwise, it is not selected as centroid. Then, the improved PSO with neighborhood search strategy and diversity mechanism is applied to find the best cluster centroids, M_a, from M through a specified number of iterations. k-means is then employed to refine the found center centroids. Similar to DCPSO [9] and DCPG [26], after implementing of k-means, M is set to $M_a \cup M_b$, where M_b is randomly chosen set of centroids from data set S. The algorithm is then repeated using new M. When the terminal criteria is met, M_a will be the resulting "optimum" set of cluster centroids, and the number of elements of M_a will be the "optimum" number of clusters in data set S. The main steps of DCPSONS algorithm are summarized in Table 1. In Table 1, two loops called inner loop and outer loop are structured, the main purpose of inner loop is to find the number of clusters and initial state for k-means to refine the cluster centroids by using the modified binary PSO algorithm, while the outer loop including the inner loop is to implement k-means algorithm on the global best solution *gbest* found from the inner loop. The loop number of inner and outer is experimentally defined. In particular, steps from 1 to 4 are to set the parameter values and initialize the population, the fitness values of particles in population are implemented at steps from 5 to 8, the binary PSO process is implemented at steps of 11 and 12, steps from 13 to 15 are the process of diversity mechanism, and steps from 18 to 28 are to implement the neighborhood search strategy process.

3.1 Validity Indices

Similar to [26, 9], in our proposed dynamic clustering approach the distances within each cluster and the distances between clusters were considered in the measurement index, which was proposed by [41] as follows:

$$VI = (c \times N(0,1) + 1) \times \frac{intra}{inter} \tag{16}$$

where $(c \times N(0,1) + 1)$ is regarded as a punishment value to avoid having too few clusters, c is a constant set to 30, and $N(0,1)$ stands for the Gaussian function of the number of clusters with mean of zero and standard deviation of one. Turi [41] indicated that dynamic clustering results falling in the interval of [2, maximum number of clusters]. The *intra* term is the average of all the distances between each data point and its cluster centroid m_k is defined as follows:

$$intra = \frac{1}{N_p} \sum_{k=1}^{K} \sum_{u \in C_k} \|u - m_k\|^2. \tag{17}$$

Its purpose is to calculate the intensity of intra-clusters. The approach is to calculate the Euclidean distance of the data point and the center of cluster, sum up all the shortest distance of each data point and the center of cluster, and then divide by the total data tuples (N_p). If the *intra* value is smaller (larger), the clustering efficiency for algorithm is considered better (worse). Lastly, the *inter* is the distance between two clusters and the formula is shown as Equation (18).

$$inter = \min\{\|m_k - m_{kk}\|^2\} \tag{18}$$

where $\forall k = 1, 2, \ldots, K-1$ and $kk = k+1, \ldots, K$.

```
1  Initialize parameters of the algorithm;
2  Generate randomly $N_c$ centers of clusters to form the pool of cluster centroids $M$
   from data set $S$;
3  Initialize the swarm $P$ with the position of particle $x_{ik} \sim U(0,1)$, $i = 1, 2, .., NP$,
   $k = 1, 2, .., N_c$ using Equation (15);
4  Randomly initialize the velocity of each particle in $S$ such that $v_{ik} \in [-V_{max}, V_{max}]$;
   /*Calculate the fitness of particle*/
5  For each particle in $P$ do
6    Partition data based on the centroids shown in the particle by assigning each
     data point to the closest (in term of Euclidean distance) cluster;
7    Calculate the fitness value of each particle according to Equation (16);
8  End
9  Update $pbest$ and $gbest$;
10 For each particle in $P$ do
11   Update the velocity and position of particle according to Equations (3) and (14),
     respectively;
12   Calculate the fitness value of particle similar to steps of 3, 4;
     /*Diversity mechanism*/
13   Generate a new trial particle $TP_i$ by Equations (10), (11);
14   Calculate the fitness value of $TP_i$;
15   Select a fitter one between $P_i$ and $TP_i$ as the new $P_i$ by Equation (12);
16   Update $pbest$ and $gbest$;
17 End
   /* Neighborhood search strategy */
18 For each particle in $P$ do
19   If $rand(0,1) \leq P_{ns}$ then
20     Select the best particle $nbest$ from the local neighborhood of current particle
       ($i^{\text{th}}$ particle);
21     Generate a trial particle $L_i$ according to Equations (6), (9), the position of particle is then changed to a bit of $\{0,1\}$ by using Equation (14) on its position;
22     Calculate the fitness value of $L_i$;
23     Generate a trial particle $G_i$ according to Equations (7), (8), the position of particle is then changed to a bit of $\{0,1\}$ by using Equation (14) on its position;
24     Calculate the fitness value of $G_i$;
25     Select the best one among $P_i$, $L_i$, $G_i$ as the new $P_i$;
26   End
27   Update $pbest$ and $gbest$;
28 End
29 Return to Step 10 until the pre-specified number of iterations is satisfied
   (called inner loops);
30 Refine the best cluster centroids $M_a$ by applying $k$-means
   where $M_a$ is formed according to $gbest$ and the pool of cluster centroids $M$;
31 Randomly reinitialize $M_b$ from data set $S$ where $M = M_a \cup M_b$;
32 Return Step 10 until termination criteria are met (called outer loops);
```

Table 1: The main steps of DCPSONS

3.2 Time Complexity

The time complexity of DCPSONS is based on the complexity of three main processes, namely, the binary PSO with diversity mechanism (steps of 10 to 17), the binary PSO with neighborhood search (steps of 18 to 28), and applying k-means process. Assuming T_1 if the number of iterations taken by the PSO to converge (step 29 of the algorithm), and T_2 is the number of iterations taken by DCPSONS to converge (step 32 of the algorithm).

In the first process of the binary PSO with diversity mechanism, the complexity consists of the complexity of partitioning of X and calculating the quality of partition. So, the complexity of this process will be $O(T_1 \times T_2 \times (NP + p_r \times NP) \times N_c \times N \times D)$, where the diversity process is $O(T_1 \times T_2 \times (p_r \times NP) \times N_c \times N \times D)$. So, the complexity of this process will be $O(T_1 \times T_2 \times NP \times N_c \times N \times D)$.

Similar to the first process, the complexity of the second process is $O(T_1 \times T_2 \times (p_{ns} \times N_c \times NP) \times N \times D)$. So, it will be $O(T_1 \times T_2 \times NP \times N_c \times N \times D)$.

In the third process, the complexity of k-means algorithm in T_1 iterations is $O(T_1 \times N)$. From the complexity of three processes, the complexity of the algorithm will be $O(T_1 \times T_2 \times NP \times N_c \times N \times D)$. The parameters T_1, T_2, N_c, NP, D can be fixed in advance, and T_1, T_2, N_c, NP, $D \ll N$. Let ς be the multiplication of T_1, T_2, N_c, NP, D (i.e. $\varsigma = T_1 \times T_2 \times N_c \times NP \times D$). If $\varsigma \ll N$ then the time complexity of the algorithm will be $O(N)$. However, if $\varsigma \approx N$ then it will be $O(N^2)$.

4 EXPERIMENTS FOR DYNAMIC CLUSTERING

To evaluate the performance of DCPSONS algorithm, in this section the simulation results will be shown and analysed. In this case, we use the benchmark data sets described, four of them were used in [26], to test and compare to some dynamic clustering algorithm of DCPSO [9] and DCPG [26]. Because DCPSO and DCPG also employ the binary PSO to find out the best number of clusters through a specified number of iterations and the results are then refined by k-means, and DCPSONS is improved from their ideas. On the other hand, in the literature of presenting DCPSO and DCPG, several other dynamic clustering algorithms were compared to.

4.1 Benchmark Data Sets

In order to evaluate the performance of the proposed DCPSONS algorithm, artificial data sets and real-world data sets are used, the details of properties are described in Table 2 [34] and Table 3 [35]. Note that:

1. in Table 2, just Dataset1 has no overlapping clusters,
2. the data sets shown in Table 3 are used as a benchmark for the evaluation of clustering methods in recent years.

In addition, the data set sonar has the number of features that the complexity of the algorithm is $O(N^2)$ (described in Section 3.2), and the data set pendigits is a high size big data set.

	Data Set	Size	Features	Number of Clusters
1	Dataset1	400	3	4
2	Dataset2	250	2	5
3	Dataset3	300	2	6
4	Dataset4	500	2	10

Table 2: The main properties of artificial data sets used in [27, 24, 36, 37]

	Data Set	Size	Features	Number of Clusters
1	Iris	150	4	3
2	Wine	178	13	3
3	Glass	214	10	6
4	Ecoli	336	7	8
5	Liver disorder	345	6	2
6	Vowel	871	3	6
7	Vowel2	528	10	11
8	Pima	768	8	2
9	WDBC	569	30	2
10	CMC	1 473	10	3
11	Sonar	208	60	2
12	Pendigits	10 992	16	10

Table 3: The main properties of real-world data sets used in [27, 26, 22, 7]

4.2 Parameters Setting

The parameters of DCPSONS algorithm were empirically set as in Table 4, and all of algorithms run over 20 times, the parameters of two other algorithms DCPSO and DCPG were set according to their experiments. For fair comparison, in this test all of competitive algorithms have the same total number of training (*MaxFEs*).

The parameter of the maximum number of clusters must initially be set, after which, the optimal solution can be found by dynamic clustering. Zhang et al. [42] suggested that the maximum number of clusters should not exceed the square root of the number of data. Thus, the maximum number of clusters was set to the rounded number of square root of the number of its data.

4.3 Results and Analysis

As above description, tests of each clustering algorithms were implemented over 20 times on all of sixteen benchmark data sets which consist of four artificial data sets and twelve real-world data sets. The average and standard deviation of fitness

Description of Parameter	Value
MaxFEs (inner × outer loops)	500 (100 × 5)
Population size (NP)	20
Inertia weight (w)	0.72
Learning factor (c_1, c_2)	$c_1 = c_2 = 1.49$
V_{max}	5
P_{ini}	0.75
P_r	0.9
P_{ns}	0.6

Table 4: The setting of parameter values for DCPSONS algorithm

values over 20 tests were used as the experimental results. Through the number of clusters and *VI* value as the measurement indexes with the average and deviation values recorded, the performance of competitive algorithms will be assessed.

The results of clusters numbers of competitive algorithms on artificial and real-world data sets are shown in Tables 5 and 6, respectively, where Avg stands for Average, SD stands for Standard Deviation (no loss of generality, in the results if the standard deviation value is less than 10^{-8}, it is considered as zero), where the best value is written in bold. The *VI* measures of competitive algorithms on artificial and real-world data sets are in Tables 7 and 8, respectively. The *VI* value indicates that the training process of the algorithm reaches convergence. Regarding the number of clusters, the algorithm, which has results of the cluster number found be closer to the actual number of clusters of the benchmark data set, is better. The convergence curves of algorithms are illustrated in Figure 2 on the artificial data sets, and Figure 3 on real-world data sets. In order to compare the performance of multiple algorithms on the test suite, we conduct Friedman test according to the suggestions of [44, 45]. Table 9 shows the average ranking of DCPSO, DCPG, and DCPSONS on fourteen data sets. The highest ranking, which belongs to DCPSONS in term of convergence speed, is written in bold.

From the tables of results and figures of convergence it is clear that on the artificial data sets, the performance of DCPSONS is better than DCPSO and DCPG in majority of data sets for finding the best number of clusters. But, on Dataset4, the found numbers of clusters of all of competitive algorithm are quite far from the actual number, the best is DCPSONS of 7.70, while the actual is 10. In terms of fitness value *VI*, the best results belongs to DCPSONS, but one belongs to DCPG, Dataset2. In general, three algorithms has the same results in Dataset1 (which is the non-overlap data set), and the convergence speed of DCPSONS is the fastest and smallest of *VI* value.

On real-world data sets, DCPSONS has the best performance of finding the best number of clusters on nine data sets, and three other data sets belong to DCPG and DCPSO algorithms. From Table 8, it can be seen that DCPSONS has the best convergence on all of data sets. Despite DCPSONS is worse in finding number of clusters of Vowel2, Pima, CMC data sets, but DCPSONS significantly different from other algorithms. From Figures of convergence curves, the proposed DCPSONS algorithm has faster convergence speed on majority of data sets, but only on Ecoli data set DCPSONS was slower at first, and at last it was better.

Data Set	Actual Number of Clusters	DCPSO Avg SD	DCPG Avg SD	DCPSONS Avg SD
Dataset1	4	**4.00**	**4.00**	**4.00**
		0	**0**	**0**
Dataset2	5	5.05	**5.00**	**5.00**
		0.22	**0**	**0**
Dataset3	6	5.90	6.15	**6.00**
		0.31	0.37	**0.00**
Dataset4	10	6.75	6.95	**7.70**
		1.12	1.28	**1.49**

Table 5: Comparison of clusters number for artificial data sets

Regarding the computational time, though the DCPSONS algorithm is more complicated than DCPSO, but quite simpler than DCPG, hence DCPSONS needs less than DCPG. In particular, DCPG costs much time in the performance of elitist selection for population 1 and population 2 to generate the next iterative population. The results of computational time on benchmark datasets are expressed in Table 10. From the results in Table 10, it is clear that DCPSO consumes less time than others, but DCPG costs little more than the proposed DCPSONS algorithm. However, all of competitive algorithms cost much more on big dataset (pendigits).

5 APPLICATION IN IMAGE SEGMENTATION

In this section, we will describe the results of applying the proposed algorithm in image segmentation. As mentioned above, the existing methods of image segmentation can be roughly divided into threshold, edge detection, region splitting and merging, and segmentation based on clustering algorithms. Among them, segmentation methods based on clustering algorithms are to partition the similar regions of an image into one class as much as possible, and divide the dissimilar regions into different categories, through certain criteria. In the proposed method, like to dynamic clustering the best number of regions (clusters) of image is automatically found by binary PSO with diversity and neighborhood search, each pixel is then grouped into the closest region by k-means clustering algorithm.

To demonstrate the performance of algorithm, we apply this proposed approach to perform pixel clustering with several synthetic images, and grey natural images. The parameters were set similar to Section 4.2, except the *MaxFEs* is set to 100 (50×2), and the value V_{max} is set to 255 because the range of grey value of image is from 0 to 255. The algorithm is compared to two other competitive algorithm DCPSO [9] and DCPG [26], as for dynamic clustering of above description which

Data Set	Actual Number	DCPSO	DCPG	DCPSONS
	of Clusters	Avg	Avg	Avg
		SD	SD	SD
Iris	3	2.95	**3.00**	**3.00**
		0.22	**0**	**0**
Wine	3	3.50	3.55	**3.25**
		0.76	0.69	**0.44**
Glass	6	5.45	5.55	**5.70**
		0.60	0.60	**0.80**
Ecoli	8	5.40	6.25	**6.50**
		2.68	2.02	**1.89**
Liver disorder	2	**2.30**	2.35	**2.30**
		0.47	0.49	**0.47**
Vowel	6	6.20	**6.05**	**6.05**
		0.83	**1.00**	**1.10**
Vowel2	11	6.75	**12.00**	12.05
		2.10	**1.92**	2.09
Pima	2	2.40	2.20	**2.25**
		0.50	0.41	**0.55**
WDBC	2	2.85	2.85	**2.80**
		0.67	0.67	**0.41**
CMC	3	**3.00**	**3.00**	3.10
		0.32	**0.46**	0.31
Sonar	2	3.05	3.12	**2.86**
		0.45	0.58	**0.37**
Pendigits	10	7.42	9.18	**9.25**
		1.42	0.89	**0.65**

Table 6: Comparison of clusters number on real-world data sets

Data Set	DCPSO	DCPG	DCPSONS
	Avg	Avg	Avg
	SD	SD	SD
Dataset1	**0.0522**	**0.0522**	**0.0522**
	0	**0**	**0**
Dataset2	0.1824	**0.1621**	0.1664
	0.0815	**0.0496**	0.0704
Dataset3	0.0648	0.05515	**0.0551**
	0.0299	0	**0.00007**
Dataset4	0.1955	0.1986	**0.1762**
	0.0334	0.0398	**0.0359**

Table 7: Comparison of *VI* value on artificial data sets

Data Set	DCPSO	DCPG	DCPSONS
	Avg	Avg	Avg
	SD	SD	SD
Iris	0.2225	0.2046	**0.1003**
	0.1477	0.0783	**0.0091**
Wine	0.0966	0.0959	**0.0956**
	0.0121	0.0113	**0.0082**
Glass	0.0177	0.1717	**0.1601**
	0.0747	0.0531	**0.0197**
Ecoli	0.3197	0.3064	**0.3060**
	0.0620	0.0456	**0.0610**
Liver disorder	0.1277	0.1180	**0.0981**
	0.0410	0.0470	**0.0274**
Vowel	0.2787	0.2855	**0.2756**
	0.0299	0.0212	**0.0291**
Vowel2	0.5255	0.5061	**0.5009**
	0.0515	0.0451	**0.0431**
Pima	0.1065	0.1167	**0.0984**
	0.0283	0.0504	**0.0239**
WDBC	0.0849	0.0816	**0.0717**
	0.0343	0.0338	**0.0219**
CMC	0.1814	0.1848	**0.1647**
	0.0209	0.0223	**0.0106**
Sonar	0.4443	0.3989	**0.3366**
	0.0311	0.0421	**0.0452**
Pendigits	0.4343	0.4139	**0.3932**
	0.0658	0.0325	**0.0532**

Table 8: Comparison of *VI* value on real-world data sets

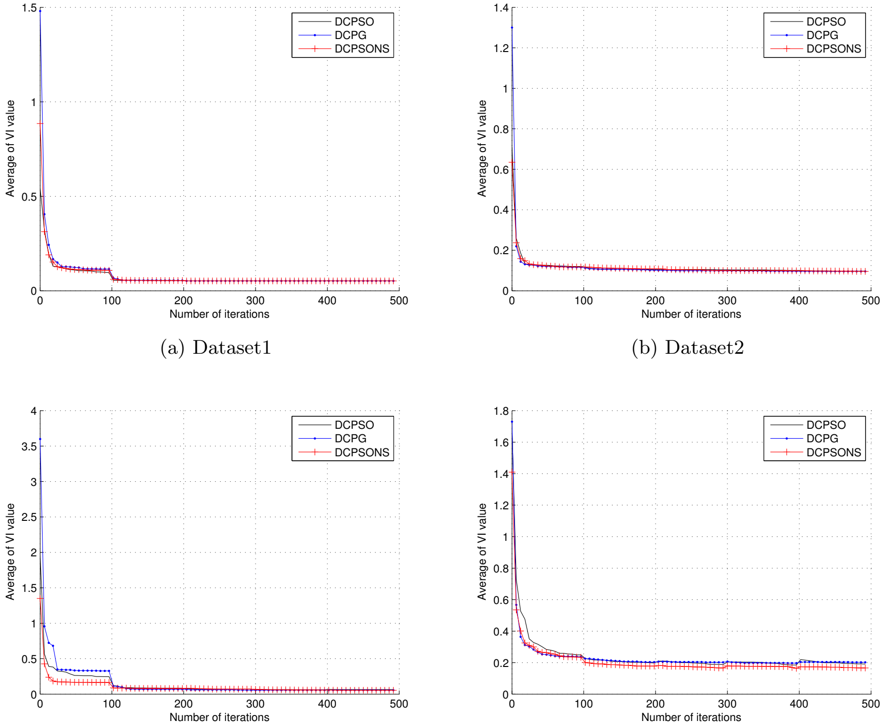
Algorithms	Rankings
DCPSO	2.64
DCPG	2.21
DCPSONS	**1.14**

Table 9: Average ranking achieved by Friedman test

are the dynamic clustering algorithm and application in image segmentation. The details of results will be describe as following sections.

5.1 Synthetic Images

The synthetic images used in this test were conducted by a tool called SIGT [43] including five images with size of 100 × 100. The histogram curves of synthetic images are illustrated in Figure 4. In this case, each image is encode as a 1-D data set, with each element being a 1-D vector containing a grey value of pixel image.



(a) Dataset1

(b) Dataset2

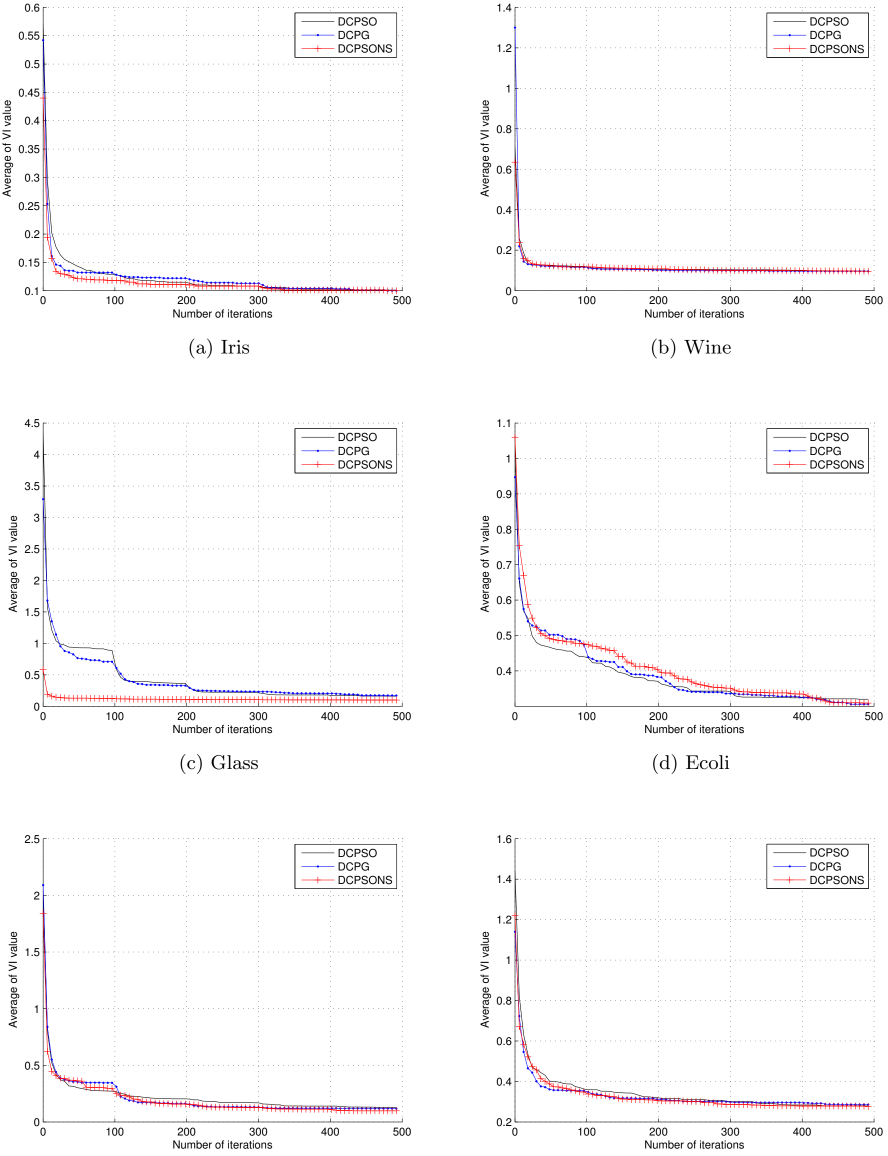
(c) Dataset3

(d) Dataset4

Figure 2: The convergence curves on four artificial data sets

Data Set	DCPSO	DCPG	DCPSONS
Iris	0.72	1.38	1.03
Wine	2.72	3.98	3.06
Glass	2.37	3.56	2.74
Ecoli	3.42	4.07	3.53
Vowel	6.70	6.42	5.79
Vowel2	7.39	11.24	9.09
Pima	11.00	11.02	10.57
WDBC	24.11	24.44	24.48
CMC	21.43	23.90	23.00
Sonar	61.29	82.56	80.84
Pendigits	842.22	1 120.10	1 025.65
Total	**986.39**	**1 296.11**	**1 192.88**

Table 10: The average of computational time (in second)



(a) Iris

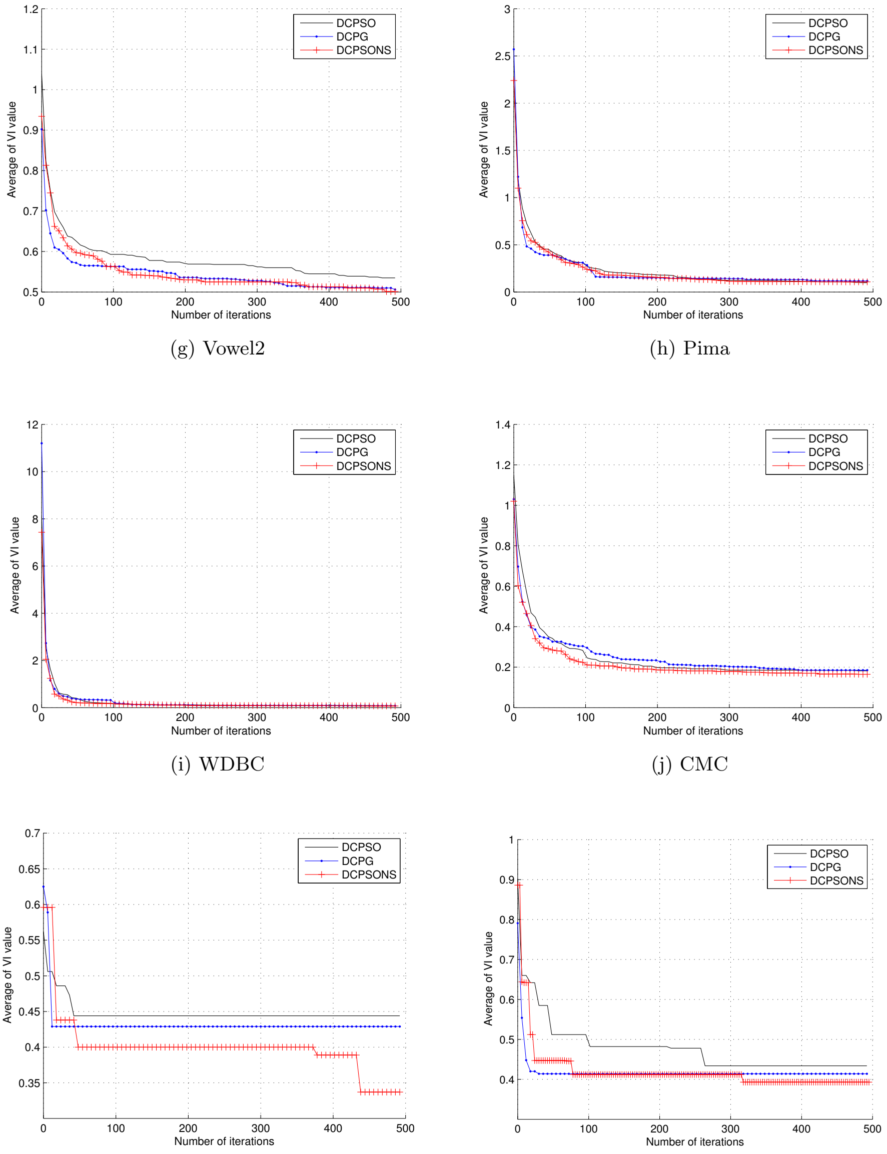
(b) Wine

(c) Glass

(d) Ecoli

(e) Liver disorder

(f) Vowel



(g) Vowel2

(h) Pima

(i) WDBC

(j) CMC

(k) Sonar

(l) Pendigits

Figure 3: The convergence curves on real-world data sets

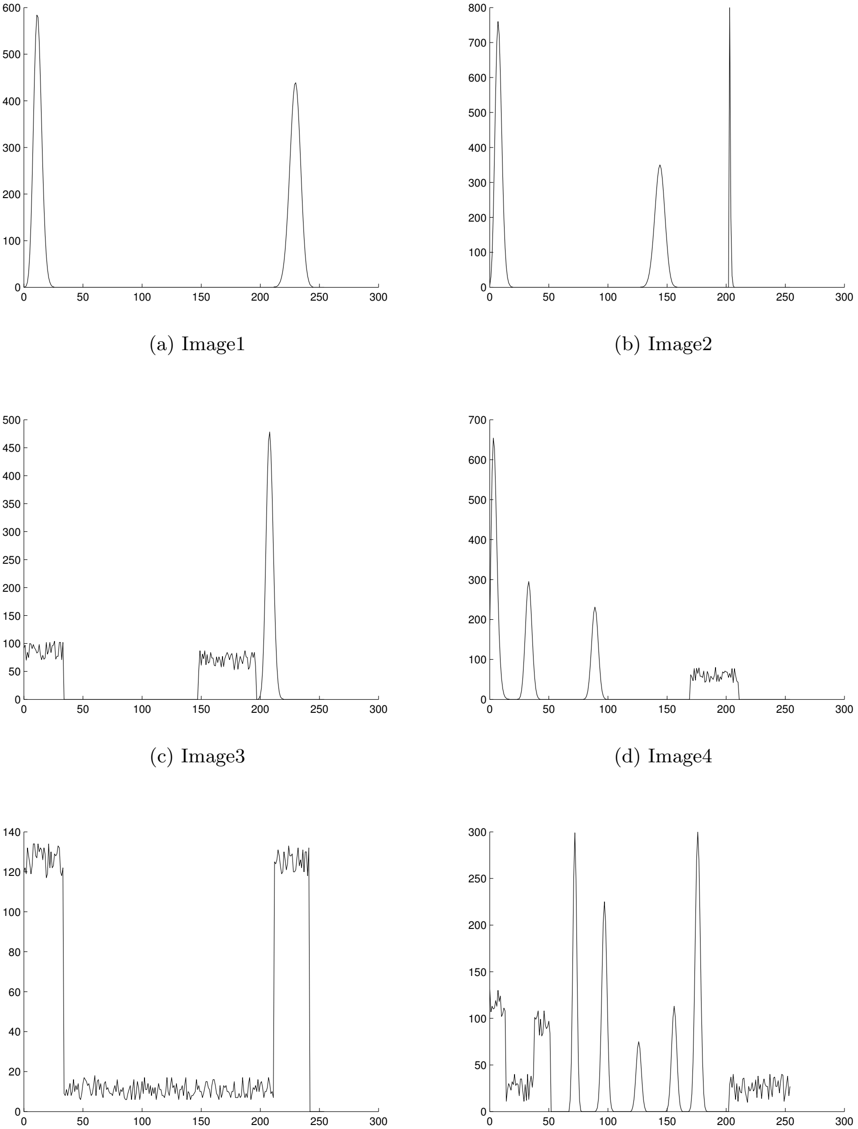
The results of three competitive algorithms on cluster number and fitness value *VI* are shown in Tables 11 and 12, respectively, where the best value is written in bold. The results in Tables 11 and 12 show that three competitive algorithms have generally performed well, especially DCPSONS which obtained the results of cluster number closer actual actual number than others. In terms of fitness value *VI*, the proposed has the best results in majority of data sets, while Image4 belongs to DCPSO algorithm. In particular case, the found cluster number of Image4 is five different from actual number, because the fourth cluster has length of grey uniform distribution quite longer than others, so this regions is easy to be segmented into two or more clusters. And more, in case of Image5, the actual number clusters is 3, but the found results are different, because the second cluster, which has long length of grey uniform distribution, may be segmented into more than actual number. Whereas in case of Image3, all of three algorithms merged two clusters of the second one and the third one into a cluster.

Image	Actual Number	DCPSO	DCPG	DCPSONS
	of Clusters	Avg	Avg	Avg
		SD	SD	SD
Image1	2	2.00	2.00	2.00
		0	0	0
Image2	3	3.00	3.00	3.00
		0	0	0
Image3	3	2.00	2.00	2.00
		0	0	0
Image4	4	5.00	5.00	5.00
		0	0	0
Image5	3	5.00	4.80	4.76
		0.85	1.03	1.20
Image6	9	6.89	6.67	7.20
		1.88	2.00	1.86

Table 11: Comparison of cluster number on synthetic images

5.2 Natural Images

For evaluating the performance of algorithm on natural image, four grey natural images were used in the evaluation, namely Lena, mandrill, peppers, and MRI. The original images and its histograms are illustrated in Figure 5. Like [9, 41], the optimal number of clusters of each image was suggested and is shown in Table 13. The results of found cluster number and fitness value *VI* are in Tables 13 and 14, respectively, where the best value is written in bold, and the segmented images of three algorithms are shown in Figure 5. The results listed in the tables and shown in the figure demonstrate that the proposed algorithm obtained better than others on almost test images in terms of fitness value *VI*, number of clusters, and quality of segmented image.



(a) Image1

(b) Image2

(c) Image3

(d) Image4

(e) Image5

(f) Image6

Figure 4: The histograms of synthetic images

Image	DCPSO	DCPG	DCPSONS	
	Avg	Avg	Avg	
	SD	SD	SD	
Image1	**2.08**	**2.08**	**2.08**	($\times 10^{-13}$)
	0	**0**	**0**	
Image2	0.0245	**0.0244**	**0.0244**	
	0.0002	**0.0002**	**0.0002**	
Image3	**1.16**	**1.16**	**1.16**	($\times 10^{-13}$)
	0	**0**	**0**	
Image4	**0.0261**	0.0280	0.0262	
	0.0027	0.0051	0.0017	
Image5	0.111	0.117	**0.108**	
	0.045	0.068	**0.046**	
Image6	0.0823	0.0775	**0.0708**	
	0.0223	0.0163	**0.0184**	

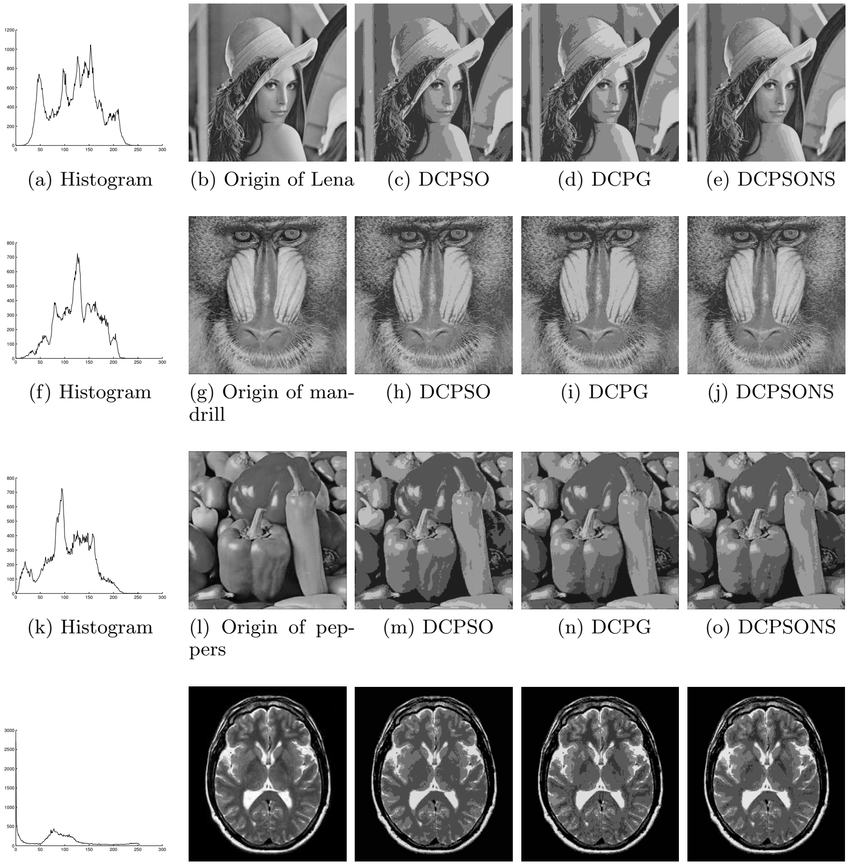
Table 12: Comparison of *VI* value on synthetic images

Image	Optimal Range	DCPSO	DCPG	DCPSONS
		Avg	Avg	Avg
		SD	SD	SD
Lena	5–10	5.80	6.17	6.75
		0.84	0.98	2.22
Mandrill	5–10	5.67	6.12	6.25
		0.58	1.73	1.02
Peppers	6–10	6.25	5.33	6.12
		1.12	0.58	1.23
MRI	4–8	5.51	5.33	5.87
		0.52	0.58	0.86

Table 13: Comparison of cluster number on natural images

Image	DCPSO	DCPG	DCPSONS
	Avg	Avg	Avg
	SD	SD	SD
Lena	0.112	0.108	**0.105**
	0.007	0.012	**0.022**
Mandrill	0.138	0.117	**0.097**
	0.025	0.015	**0.005**
Peppers	**0.101**	**0.101**	0.117
	0.011	**0.016**	0.024
MRI	0.073	0.069	**0.066**
	0.021	0.019	**0.004**

Table 14: Comparison of *VI* value on natural images



(a) Histogram (b) Origin of Lena (c) DCPSO (d) DCPG (e) DCPSONS

(f) Histogram (g) Origin of mandrill (h) DCPSO (i) DCPG (j) DCPSONS

(k) Histogram (l) Origin of peppers (m) DCPSO (n) DCPG (o) DCPSONS

(p) Histogram (q) Origin of MRI (r) DCPSO (s) DCPG (t) DCPSONS

Figure 5: The original grey natural images. Test images from the first row to fourth row are Lena, mandrill, peppers, MRI, respectively

6 CONCLUSIONS

This study proposed DCPSONS algorithm is to solve the problems with the unknown number of clusters in advance applied in image segmentation. The approach consists of two phases, the first phase is to find out the optimal number of clusters based on the features of data, the second phase is to refine the chosen clusters by k-means. Fourteen benchmark data sets were used to evaluate the performance of DCPSONS in dynamic clustering. The results show that the DCPSONS algorithm can obtain the right number of clusters and achieve better clustering results. Moreover, the DCPSONS algorithm was compared to DCPSO, DCPG algorithms of dynamic clustering, based on the improved PSO with diversity mechanism and neighborhood search, the algorithm not easy to fall into the local optimal solution.

In image segmentation application, the results on synthetic and natural images show that the performance of DCPSONS is better then the other competitive DCPSO and DCPG algorithms in terms of number of clusters, convergence rate, and segmented image quality.

Overall, the results on fourteen benchmark data sets and test images demonstrate that the proposed DCPSONS algorithm is an effective, robust and efficient clustering algorithm in both of data clustering and dynamic clustering problems.

In the future, we will expand the method by using GPU and MIC architecture for big datasets and apply it to other domains.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 61070008 and 61364025).

REFERENCES

[1] Hartigan, J. A.: Clustering Algorithms. 1st Edition. Wiley, New York, 1975.

[2] Bezdek, J. C.: Pattern Recognition with Fuzzy Objective Function Algorithms. New York, Plenum, 1981, doi: 10.1007/978-1-4757-0450-1.

[3] Chattopadhyay, S.—Pratihar, D. K.—De Sarkar, S. C.: A Comparative Study of Fuzzy C-Means Algorithm and Entropy-Based Fuzzy Clustering Algorithm. Computing and Informatics, Vol. 30, 2011, No. 4, pp. 701–720.

[4] Khedr, A. M.: Nearest Neighbor Clustering over Partitioned Data. Computing and Informatics, Vol. 30, 2011, No. 5, pp. 1011–1036.

[5] Jain, A. K.—Murty, M. N.—Flynn, P. J.: Data Clustering: A Review. ACM Computing Surveys (CSUR), Vol. 31, 1999, No. 3, pp. 264–323, doi: 10.1145/331499.331504.

[6] Xu, R.—Wunsch, D.: Survey of Clustering Algorithms. IEEE Transaction on Neural Networks, Vol. 16, 2005, No. 3, pp. 645–678, doi: 10.1109/tnn.2005.845141.

[7] Neshat, M.—Yazdi, S. F.—Yazdani, D.—Sargolzaei, M.: A New Cooperative Algorithm Based on PSO and k-Means for Data Clustering. Journal of Computer Science, Vol. 8, 2012, No. 2, pp. 188–194, doi: 10.3844/jcssp.2012.188.194.

[8] Zhang, J.—Wang, Y.—Feng, J.: A Hybrid Clustering Algorithm Based on PSO with Dynamic Crossover. Soft Computing, Vol. 18, 2013, No. 5, pp. 961–979.

[9] Omran, M. G. H.—Salman, A.—Engelbrecht, A. P.: Dynamic Clustering Using Particle Swarm Optimization with Application in Image Segmentation. Pattern Analysis and Applications, Vol. 8, 2006, No. 4, pp. 332–344, doi: 10.1007/s10044-005-0015-5.

[10] Kennedy, J.—Eberhart, R. C.: Particle Swarm Optimization. Proceedings of IEEE International Conference on Neuron Networks, Perth, Australia, Vol. 4, 1995, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.

[11] Wang, H.—Sun, H.—Li, C.—Rahnamayan, S.—Pan, J.-S.: Diversity Enhanced Particle Swarm Optimization with Neighborhood Search. Information Sciences, Vol. 223, 2013, pp. 119–135, doi: 10.1016/j.ins.2012.10.012.

[12] Tran, D. C.—Wu, Z.—Wang, H.: A Novel Enhanced Particle Swarm Optimization Method with Diversity and Neighborhood Search. Proceedings of 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2013), 2013, pp. 180–187, doi: 10.1109/SMC.2013.38.

[13] Martin, D.—Fowlkes, C.—Tal, D.—Malik, J.: A Database of Human Segmented Natural Images and Its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. Proceedings of 8th IEEE International Conference on Computer Vision (ICCV 2001), 2001, pp. 416–423, doi: 10.1109/iccv.2001.937655.

[14] Das, S.—Abraham, A.—Konar, A.: Automatic Clustering Using an Improved Differential Evolution Algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. 38, 2008, No. 1, pp. 218–237.

[15] Kwedlo, W.: A Clustering Method Combining Differential Evolution with the *k*-Means Algorithm. Pattern Recognition Letters, Vol. 32, 2011, No. 12, pp. 1613–1621, doi: 10.1016/j.patrec.2011.05.010.

[16] Shelokar, P.—Jayaraman, V. K.—Kulkarni, B. D.: An Ant Colony Approach for Clustering. Analytica Chimica Acta, Vol. 509, 2004, No. 2, pp. 187–195, doi: 10.1016/j.aca.2003.12.032.

[17] Zhang, C.—Ouyang, D.—Ning, J.: An Artificial Bee Colony Approach for Clustering. Expert Systems with Applications, Vol. 37, 2010, No. 7, pp. 4761–4767.

[18] Kao, Y.-T.—Zahara, E.—Kao, I-W.: A Hybridized Approach to Data Clustering. Expert Systems with Applications, Vol. 34, 2008, No. 3, pp. 1754–1762.

[19] Das, S.—Abraham, A.—Konar, A.: Automatic Kernel Clustering with a Multi-Elitist Particle Swarm Optimization Algorithm. Pattern Recognition Letters, Vol. 29, 2008, No. 5, pp. 688–699, doi: 10.1016/j.patrec.2007.12.002.

[20] Tsai, C.-Y.—Kao, I-W.: Particle Swarm Optimization with Selective Particle Regeneration for Data Clustering. Expert Systems with Applications, Vol. 38, 2011, No. 6, pp. 6565–6576, doi: 10.1016/j.eswa.2010.11.082.

[21] Cura, T.: A Particle Swarm Optimization Approach to Clustering. Expert Systems with Applications, Vol. 39, 2012, No. 1, pp. 1582–1588, doi: 10.1016/j.eswa.2011.07.123.

[22] Jiang, B.—Wang, N.—Wang, L.: Particle Swarm Optimization with Age-Group Topology for Multimodal Functions and Data Clustering. Communications in Nonlinear Science and Numerical Simulation, Vol. 18, 2013, No. 11, pp. 3134–3145.

[23] Gliwa, B.—Byrski, A.: Hybrid Neuro-Fuzzy Classifier Based on NEFCLASS Model. Computer Science, Vol. 12, 2011, pp. 115–135.

[24] Bandyopadhyay, S.—Maulik, U.: Nonparametric Genetic Clustering: Comparison of Validity Indices. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 31, 2001, No. 1, pp. 120–125.

[25] Rosenberger, C.—Chehdi, K.: Unsupervised Clustering Method with Optimal Estimation of the Number of Clusters: Application to Image Segmentation. Proceedings of 15th International Conference on Pattern Recognition (ICPR' 00), Vol. 1, 2000, pp. 1656–1659, doi: 10.1109/icpr.2000.905473.

[26] Kuo, R. J.—Syu, Y. J.—Chen, Z.-Y.—Tien, F. C.: Integration of Particle Swarm Optimization and Genetic Algorithm for Dynamic Clustering. Information Sciences, Vol. 195, 2012, pp. 124–140, doi: 10.1016/j.ins.2012.01.021.

[27] Masoud, H.—Jalili, S.—Hasheminejad, S. M. H.: Dynamic Clustering Using Combinatorial Particle Swarm Optimization. Applied Intelligence, Vol. 38, 2013, No. 3, pp. 289–314, doi: 10.1007/s10489-012-0373-9.

[28] Pal, N. R.—Pal, S. K.: A Review on Image Segmentation Techniques. Pattern Recognition, Vol. 26, 1993, pp. 1277–1294, doi: 10.1016/0031-3203(93)90135-J.

[29] Fu, K. S.—Mui, J. K.: A Survey on Image Segmentation. Pattern Recognition, Vol. 13, 1981, No. 1, pp. 3–16, doi: 10.1016/0031-3203(81)90028-5.

[30] Chen, C.-Y.—Fun, Y.: Particle Swarm Optimization Algorithm and Its Application to Clustering Analysis. Proceeding of 2004 IEEE International Conference on Networking, Sensing and Control, Vol. 2, 2004, pp. 789–794, doi: 10.1109/ICNSC.2004.1297047.

[31] Clerc, M.—Kennedy, J.: The Particle Swarm – Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation, Vol. 6, 2002, No. 1, pp. 58–73, doi: 10.1109/4235.985692.

[32] Mendes, R.—Kennedy, J.—Neves, J.: The Fully Informed Particle Swarm: Simpler, Maybe Better. IEEE Transaction on Evolutionary Computation, Vol. 8, 2004, No. 3, pp. 204–210, doi: 10.1109/tevc.2004.826074.

[33] Riget, J.—Vesterstrom, J. S.: A Diversity-Guided Particle Swarm Optimizer – the ARPSO. Technical Report, EVAlife, Denmark, 2002.

[34] Bandyopadhyay, S. et al.: Artificial Data Sets for Data Mining. Available on: `http://www.isical.ac.in/~sanghami/data.html`.

[35] UCI Repository of Machine Learning Databases. Available on: `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

[36] Bandyopadhyay, S.—Maulik, U.: Genetic Clustering for Automatic Evolution of Clusters and Application to Image Classification. Pattern Recognition, Vol. 35, 2002, No. 6, pp. 1197–1208, doi: 10.1016/s0031-3203(01)00108-x.

[37] Kao, Y.—Lee, S.-Y.: Combining k-Means and Particle Swarm Optimization for Dynamic Data Clustering Problems. IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS), 2009, pp. 757–761, doi: 10.1109/icicisys.2009.5358020.

[38] Kennedy, J.—Eberhart, R.: A Discrete Binary Version of the Particle Swarm Algorithm. Proceedings of the Conference on Systems, Man, and Cybernetics, 1997, pp. 4104–4109, doi: 10.1109/icsmc.1997.637339.

[39] Van den Bergh, F.: An Analysis of Particle Swarm Optimizers. Ph.D. Thesis, Department of Computer Science, University of Pretoria, 2002.

[40] Kuncheva, L.—Bezdek, J.: Nearest Prototype Classification: Clustering, Genetic Algorithms, or Random Search? IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Vol. 28, 1998, No. 1, pp. 160–164.

[41] Turi, R. H.: Clustering-Based Color Image Segmentation. Ph.D. Thesis, Monash University, Australia, 2001.

[42] Zhang, D.—Liu, X.—Guan, Z.: A Dynamic Clustering Algorithm Based on PSO and Its Application in Fuzzy Identification. Proceedings of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP '06), 2006, pp. 232–235, doi: 10.1109/IIH-MSP.2006.264987.

[43] Salman, A.—Omran, M.—Engelbrecht, A. P.: SIGT: Synthetic Image Generation Tool for Clustering Algorithms. ICGST International Journal Graph Vision Image Process (GVIP), Vol. 2, 2005.

[44] Derrac, J.—García, S.—Molina, D.—Herrera, F.: A Practical Tutorial on the Use of Nonparametric Statistical Tests as a Methodology for Comparing Evolutionary and Swarm Intelligence Algorithms. Swarm and Evolutionary Computation, Vol. 1, 2011, No. 1, pp. 3–18, doi: 10.1016/j.swevo.2011.02.002.

[45] García, S.—Fernández, A.—Luengo, J.—Herrera, F.: Advanced Nonparametric Tests for Multiple Comparisons in the Design of Experiments in Computational Intelligence and Data Mining: Experimental Analysis of Power. Information Sciences, Vol. 180, 2010, pp. 2044–2064, doi: 10.1016/j.ins.2009.12.010.

Dang Cong Tran works for Vietnam Academy of Science and Technology, Hanoi, Vietnam. He received his B.Sc. and M.Sc. degrees in information technology from Hanoi University of Science and Technology, Hanoi, Vietnam in 1997 and 2007, respectively. He is currently pursuing his Ph.D. degree at the State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China. His current research interests include evolutionary computation, pattern recognition, multiobjective optimization, data mining, and applications of evolutionary computation.

Zhijian Wu works for the State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan, China. He received his B.Sc. degree in mathematics from Jiangxi University, China, in 1983, his M.Sc. degree from the Department of Mathematics, Wuhan University, in 1988, and Ph.D. degree from the School of Computing, Wuhan University, Wuhan, China, in 2004. Now he is Full Professor with the State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China. His current research interests include evolutionary computation, intelligent computing, parallel computing, and intelligent information processing.