Integrated PSO and DE for Data Clustering

Suresh Satapathy, Divya Maheshwari, Sai Hanuman A, Vinaya Babu A, P. K. Patra, and B. N. Biswal

***Abstract*—This paper studies the applicability of hybridization of Differential Evolution (DE) and PSO techniques to data clustering problem. A new way of integrating DE and PSO is explored in the paper. In one approach, a parallel DE and PSO developed and in other, a transitional approach of alternate DE and PSO technique followed. Simulations for number of data sets show that the proposed integrated approach provides better clustering performance.**

***Index Terms*—Clustering, PSO, differential evolution.**

I. Introduction

Data clustering is the process of grouping together similar multi-dimensional data vectors into a number of clusters or bins. Clustering algorithms have been applied to a wide range of problems, including exploratory data analysis, data mining [1], image segmentation [2] and mathematical programming [3,4]. Many clustering algorithms were proposed till date. Clustering algorithms can be grouped into two main classes of algorithms, namely supervised and unsupervised. With supervised clustering, the learning algorithm has an external teacher that indicates the target class to which a data vector should belong. For unsupervised clustering, a teacher does not exist, and data vectors are grouped based on distance from one another. The well known K-Means clustering algorithm is one of the most important clustering algorithm in which the measure of similarity is used to determine how close two patterns are to one another. The K-means clustering groups' data vectors into a predefined number of clusters, based on Euclidean distance as similarity measure. Data vectors within a cluster have small Euclidean distances from one another, and are associated with one centroid vector. Although K-means is very popular, it can be trapped in local optima values. There have been many works to overcome the demerits of K-means approach. Among them randomized methods such as genetic algorithm (GA) [6], Evolutionary Strategies (ES) and Particle Swarm Optimization (PSO)[5] are found to be very suitable.

In this paper we propose a new method of integrating PSO and Differential Evolution (DE) technique [7]. In our proposed approach the DE and PSO are integrated. Two ways are followed to use the integrated algorithm. In one approach called as "Parallel,"each algorithm run for user defined numbers of iterations simultaneously and then fixed numbers of good particles are swapped. In the other method that is called "Transitional,"one algorithm runs for user defined numbers of iterations and the results obtained pass to the other algorithm alternatively. We use the two methods to data clustering problems. To examine the performance of our proposed methods, we compare the results with K-means clustering. There are few papers to hybridize PSO and DE in literature; however the two approaches which we have adopted here seem to be new as per our knowledge

The rest of the paper is organized as follows. In Section 2 the DE is highlighted and PSO is explained in brief. The parallel and Transitional hybrid approaches are discussed in Section 3. The clustering framework is given in Section 4. Section 5 provides simulation results and Section 6 concludes with further improvements.

Manuscript received July 18, revised September 4, 2012.

The authors are with the Department of CSE, ANITS, Visakhapatnam, AP, India(e-mail: sureshsatapathy@ieee.org; sureshsatapathy@ieee.org)

II. PSO Basics and Fundamentals of DE

A. PSO

Particle swarm optimization (PSO) technique involves simulating social behavior among individuals (particles) "flying" through a multidimensional search space, each particle representing a single intersection of all search dimensions. The particles evaluate their positions relative to a goal (fitness) at every iteration, and particles in a local neighborhood share memories of their "best" positions, then use those memories to adjust their own velocities, and thus subsequent positions.

The original PSO formulae define each particle as a potential solution to a problem in D-dimensional space, with particle i represented $Xi=(xi1,xi2,...,xiD)$.Each particle also maintains a memory of its previous best position, $Pi=(pi1,pi2,...,piD)$, and a velocity along each dimension, represented as $Vi=(vi1,vi2,...,viD)$. At each iteration, the **P** vector of the particle with the best fitness in the local neighborhood, designated g, and the P vector of the current particle are combined to adjust the velocity along each dimension, and that velocity is then used to compute a new position for the particle. The portion of the adjustment to the velocity influenced by the previous best position (**P**) is considered the cognition component, and the portion influenced by the best in the neighborhood is the social component.

$$Vid(t)= w \times vid\ (t-1) + c1\times rand\,(\,) \times (pid - xid\,(t-1)) + c2\times \text{rand}\,()\times(pgd - xid\,(\text{t}-1)) \qquad (1)$$

$$xid(t)=xid(t-1)+vid(t) \qquad (2)$$

Constants c1 and c2 determine the relative influence of the social and cognitive components, and are usually both set the same to give each component equal weight as the cognitive and social learning rate [5]. Two basic variations to PSO exist based on the interpretation of the neighborhood of particles. In the lbest PSO model, the swam is divided into overlapping neighborhoods, and the best particle of each neighborhood is determined. In this model, each particle not only remember its best position (pbest) of the current particle found so far and the best positions (gbest) of the swarm found so far, but also the local best positions (nbest) of the current particle's neighbors found

B. Differential Evolution (DE)

Although PSO is attractive due to its simple concept, few parameters, and easy implementation, it also suffers from the problem of premature convergence, especially in the large scale and complex problem. And it requires many parameters tuning for obtaining optimal results. However, Differential Evolution (DE) is a parallel direct search method developed by Storn and Price in 1997 which is a population-based global optimization algorithm. It uses a real-coded representation [7]. This approach for numerical optimization is simple to implement and requires little or no parameter tuning, but gives a remarkable performance. Like all other evolutionary algorithms, the initial population is chosen randomly.

Like all other evolutionary algorithms, DE method also consists of three basic steps:

(i) Generation of population with N individuals in the d-dimensional space, randomly distributed over the entire search domain

$$\overrightarrow{X_i}(t) = [x_{i,1}(t), x_{i,2}(t), x_{i,3}(t) \dots x_{iD}(t)]$$

where t=0,1,2,….t,t+1

(ii) Replacement of this current population by a better fit new population, and

(iii) Repetition of this replacement until satisfactory results are obtained or certain criteria of termination is met.

The basic scheme of evolutionary algorithms is given below:

a) Mutation

After the random generation of population, in each generation, a Donor vector $\vec{V_i}(t)$ is created for each $\vec{X_i}(t)$.This donor vector can be created in different ways (see DE mutation schemes).

b) Recombination

Now a trial offspring vector is created by combining components from the Donor vector $\vec{V_i}(t)$ and the target vector $\vec{X_i}(t)$. This can be done in the following way

$U_{i,j}(t) = V_{i,j}(t)$ if randi,j(0,1)<=Cr

$= X_{i,j}(t)$ otherwise

where Cr is the probability of recombination.

c) Selection

Selection in DE adopts Darwinian principle "Survival Of the Fittest". Here if the trail vector yields a better fitness value, it replaces its target in the next generation; otherwise the target vector is retained in the population. Hence the population either gets better (w.r.t. the fitness function) or remains constant but never deteriorates.

$$\overrightarrow{X_i}(t+1) = \overrightarrow{U_i}(t) \text{ if } f(U_i(t)) \le f(X_i(t)),$$
$$= \overrightarrow{X_i}(t) \text{ if } f(X_i(t)) < f(U_i(t)) \quad (8)$$

DE mutation Schemes

The five different mutation schemes suggested by Price [5] is as follows:

Scheme 1-DE/rand/1

In this scheme, to create a donor vector $\vec{V_i}(t)$ for each ith member, three other parameter vectors (say the o_1, o_2, and o_3th vectors) are chosen randomly from the current population. A scalar number F is taken. This number scales the difference of any two of the three vectors and the resultant is added to the third one. For the ith donor vector, this process can be given as

$$\vec{V_i}(t+1) = \overrightarrow{X_{o_1}}(t) + F \times \left(\overrightarrow{X_{o_2}}(t) - \overrightarrow{X_{o_3}}(t)\right)$$

Scheme 2-DE/rand to best/1

This scheme follows the same procedure as that of the Scheme1. But the difference is, now the donor vector is generated by randomly selecting any two members of the population (say the $\vec{X}_{0_2}(t)$, and $\vec{X}_{0,3}(t)$ vectors) and the best vector of the current generation (say $\vec{X}_{best}(t)$). For the ith donor vector, at time t=t+1, this can be expressed as

$$\vec{V_i}(t+1) = \vec{X_i}(t) + \lambda \times \left(\vec{X}_{best}(t) - \vec{X}_i(t)\right) + F \times \left(\vec{X}_{o_2}(t) - \vec{X}_{o_3}(t)\right)$$

where λ is a control parameter in DE and ranges between [0, 2] . To reduce the number of parameters, we consider λ =F.

Scheme 3-DE/best/1

This scheme is identical to Scheme 1 except that the result of the scaled difference is added to the best vector of the current population. This can be expressed as

$$\vec{V}_i(t+1) = \vec{X}_{best}(t) + F \times \left(\vec{X}_{o_1}(t) - \vec{X}_{o_2}(t)\right)$$

Scheme 4-DE/best/2

In this scheme, the donor vector is formed by using two difference vectors as shown below

$$\vec{V}_i(t+1) = \vec{X}_{best}(t) + F \times \left(\vec{X}_{o_1}(t) - \vec{X}_{o_2}(t)\right) + F \times \left(\vec{X}_{o_3}(t) - \vec{X}_{o_4}(t)\right)$$

Scheme 5-DE/rand/2

Here totally five different vectors are selected randomly from the population, in order to generate the donor vector. This is shown below

$$\vec{V}_i(t+`) = \vec{X}_{o_1}(t) + F_1 \times \left(\vec{X}_{o_2}(t) - \vec{X}_{o_3}(t)\right) + F_2 \times \left(\vec{X}_{o_4}(t) - \vec{X}_{o_5}(t)\right)$$

Here *F*1 and *F*2 are two weighing factors selected in the range from 0 to 1. To reduce the number of parameters we may choose *F*1 = *F*2 = *F*.

The experiment we conducted in this study uses Scheme 1-DE/rand/1

Procedure for DE

1. Randomly initialize the position of the particles
2. Evaluate the fitness for each particle
3. For each particle, create Difference-Offspring

4. Evaluate the fitness of the Difference-Offspring
5. If an offspring is better than its parent then replace the parent by offspring in the next generation;
6. Loop to step 2 until the criterion is met, usually a sufficiently good fitness or a maximum number of iterations.

III. Integrated PSO-DE Algorithm

PSO and DE are much similar in their inherent parallel characteristics, both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population. Both update the population and search for the optimum with random techniques. Both systems do not guarantee success but experiments show that they have their specific advantages when solving different problems. The new hybrid technique consists in a strong co-operation of DE and PSO, since it maintains the integration of the two techniques for the entire run of simulation, Doing so, the problem of premature convergence of the best individuals of the population to a local optimum, one of the most known drawbacks found in tests of hybrid global-local strategies, has been cancelled. This paper proposes two novel approaches based on the above-mentioned idea.

A. *Ipsodetea*

The main idea of the first proposed algorithm called Integrated-PSO-DE-Transitional evolutionary algorithm (IPSODETEA) is to integrate PSO and DE methods shown in Fig. 1. The algorithm runs PSO for some time and then makes a transition to DE and it runs in DE mode for some time and then transits back to PSO. Initially the algorithm performs for n1 iterations using PSO after which all the particles in the population are selected and passed to DE. Then DE runs for n1 iterations and the whole population is then transmitted back to PSO. This process of transitional effect continues until the termination criteria are met or best results are obtained.

B. *Ipsodepea*

The second method called Integrated PSO-DE parallel evolutionary algorithm (IPSODEPEA) is introduced briefly in this section shown in Fig 2. The main idea of the hybrid algorithm is to integrate PSO and DE methods in parallel. First we initialize the population and then the population is divided into two parts and it is evolved with the two techniques respectively .The algorithm executes the two systems simultaneously and selects user specified number of best individuals (η) from each system for exchanging after a designated number of iterations (ϒ). The individual with larger fitness value has more opportunities of being selected. The worst η population of DE will be replaced by best η number of populations of PSO and Vice Versa. This process is similar to boosting process i.e. we boost both the algorithm by exchanging the best individuals with one another. The selected individuals from PSO and DE subsystems should be encoded and decoded respectively before the exchanging operations. Shown in Fig. 2 below (IPSODEPEA)

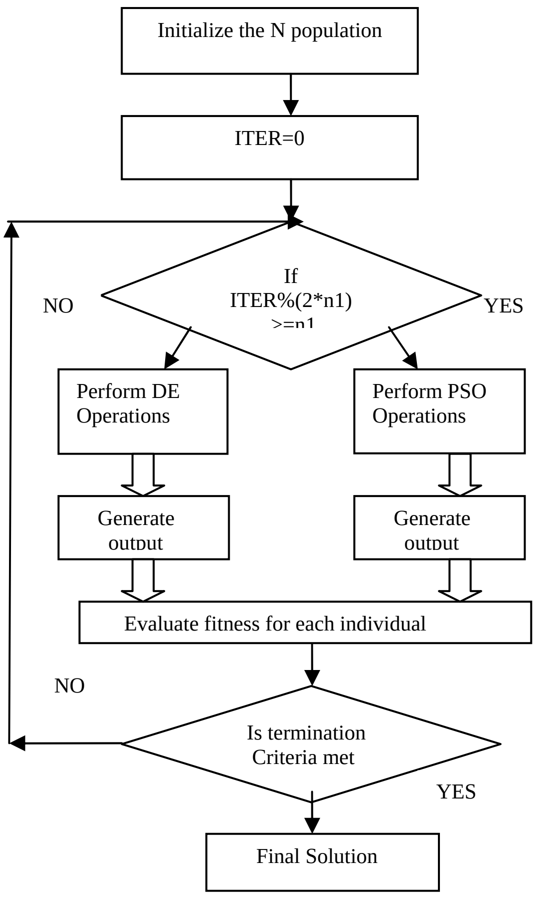

Fig. 1. IPSODETEA

IV. Clustering Framework

The following symbols are defined for the purpose of explaining the clustering in our paper

- N_d : the input dimension, i.e. the number of parameters of each data vector
- N_o : the number of data vectors to be clustered
- N_c : the number of cluster centroids (as provided by the user), i.e. the number of clusters to be formed
- z_p : the pth data vector
- m_j : the centroid vector of cluster j
- n_j : the number of data vectors in cluster j
- C_j : the subset of data vectors that form cluster *j*

The both integrated IGA-PSO parallel approach and transitional approach maintains a population where each individual population represents a potential solution to an optimization problem. In the context of clustering, an individual population represents the Nc cluster centroid vectors. That is, each individual in the population that is xi is constructed as follows:

$$x_i = \left(m_{i1}, m_{i2} \ldots\ldots\ldots m_{ij} \ldots\ldots\ldots m_{iN_c}\right)$$

where m_{ij} is the jth cluster centroid vector of the ith individual in the population in cluster c_{ij} . The fitness of each individual population is measured by the two different approaches, one by calculating the quantization error and other by the computation of intra and inter- cluster distances. The quantization error is given by

$$Q_e = \frac{\sum_{j=1}^{N_c}\left[\sum_{\forall z_p \in c_{ij}} \frac{d(z_p, m_j)}{\mathrm{mod}(c_{ij})}\right]}{N_c} \qquad (4)$$

where d is defined as the Euclidian distance between each data vector and the centroid of the cluster and is given by

$$d(z_p, m_j) = \sqrt{\sum_{k=1}^{N_d}(z_{pk} - m_{jk})^2} \qquad (5)$$

$\mathrm{mod}(c_{ij})$ is the number of data vectors belonging to cluster c_{ij} i.e. the frequency of that cluster. The other fitness function is given as

$$fit = \min(d_{intra} + \frac{1}{d_{inter}}) \qquad (6)$$

The objective is to improve the compactness of each cluster by minimizing the intra-cluster distances and improving the separation among clusters by maximizing the inter-cluster distance. In this paper we have taken equation (6) as the fitness function.

Initialize each particle to contain Nc, randomly selected cluster centroids. Here the first particle is initialized with the centroids obtained from K-means.

2. For t=1 to t_{max} do /*(t_{max} is maximum iterations)
   (a) For each particle i do
   (b) For each data vector z_p

i) Calculate the Euclidean distance $d(z_p, m_{i,j})$ to all cluster centroids C_{ij} .

ii) Assign z_p to cluster C_{ij} such that $d(z_p, m_{ij}) = \min \forall c=1,\ldots,N_c \{d(Z_p, m_{ic})\}$

iii) Calculate the fitness using equation (6).

Data set	Algorithm	Q_e	Std(Q_e)	Fitness	Std(fitness)
IRIS	K-MEANS	0.625	1.4989e-06	101.07	1.9501e-04
	IPSODETEA	0.620	0.0576	97.272	0.1956
	IPSODEPEA	0.620	0.0100	97.814	0.0549
WINE	K-MEANS	106.07	0.0010	17839	0.076
	IPSODETEA	97.018	0.1147	16431.180	0.076
	IPSODEPEA	97.586	0.0112	16540	0.1042
BREAST CANCER	K-MEANS	5.2667	9.3622e-016	3059.6	4.793e-013
	IPSODETEA	5.253	0.0277	3047.9	0.3650
	IPSODEPEA	5.1017	0.1374	3028.3	0.1600
HAYES	K-MEANS	11.296	1.324e-015	1601.7	2.397e-013
	IPSODETEA	11.127	0.0802	1487.2	1.6649
	IPSODEPEA	11.223	0.0031	1485.9	0.4267
DIABETES	K-MEANS	77.111	0.0001	50998	7.69e-012
	IPSODETEA	68.561	1.1411	49155	2.0737
	IPSODEPEA	70.787	0.1669	49182	3.6203

(c) Update the cluster centroids during evaluating the fitness function (6) using integrated PSO-DE techniques

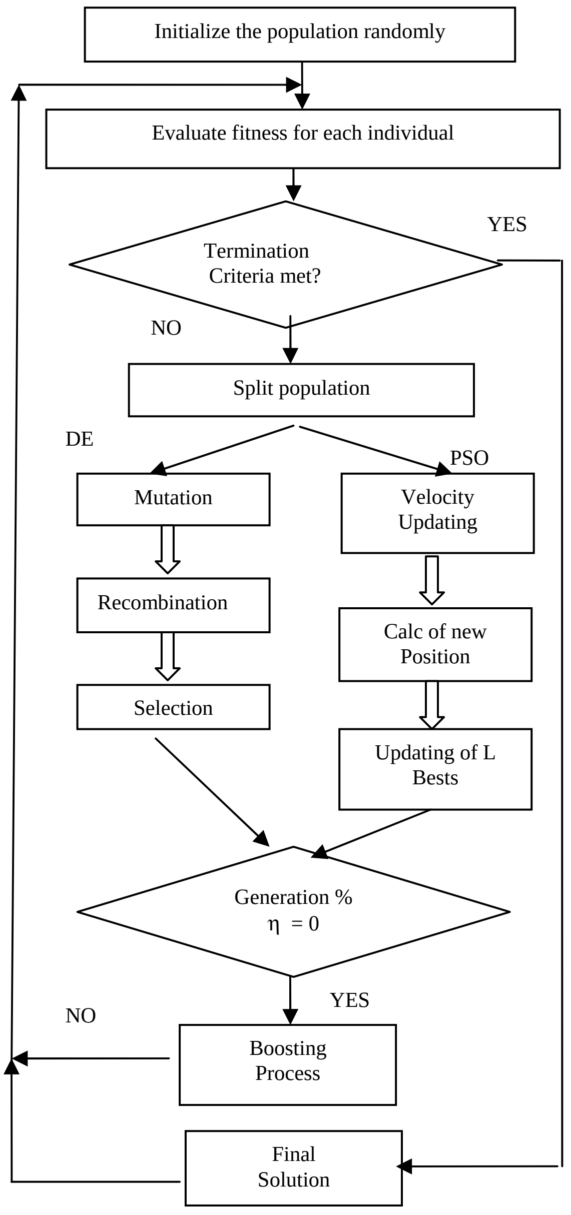

Fig. 2. IPSODEPEA

V. Simulation Results

For comparing the performances of our proposed methods we have used few data clustering problems namely, Iris, Wine, Breast Cancer, Hayes Roth and Diabetes. (Collected from UCI machine repository).

The value of F and Cr are chosen to be 0.5 and 0.9 respectively as in [7] for running DE. In PSO we have chosen c1=c2=1.60 and w= 0.72 for obtaining best results. The population size is chosen to be 10 and n1= 20, η=2 and ϒ=10. The entire algorithm is run for 200 iterations.

It is found that K-means fall in local optima for all data sets except for Iris in which it gives somewhat better result. The average results of 10 simulations runs are given in table1. The quantization error and its standard deviation define the quality of each algorithm where as fitness value and its standard deviation reports the overall clustering performance of each algorithm. The standard deviation values indicate the appropriateness of the proposed parallel and transitional method of data clustering. For Iris dataset all 3 algorithms provide better results with regard to quantization error however fitness value for k-means indicate that it fall in local optima. The cluster results (i.e. number of data vectors in each cluster) is accurate in IPSODETEA and IPSODEPEA for iris dataset. It is observed from the table that our proposed approaches are providing better fitness values. However, sometimes-parallel approach works better than transitional algorithm and vice-versa.

VI. Conclusion and Future Improvement

We have proposed a new method of integrating PSO and DE . In our proposed approach the DE and PSO are integrated in parallel and transitional way. In one approach called as "Parallel," each algorithm like DE and PSO run for user defined numbers of iterations simultaneously and fixed numbers of good particles are swapped. In the other method that is called "Transitional," one algorithm runs for user defined numbers of iterations and the results obtained passed to the other algorithm alternatively. We used the two methods to data clustering problems. The simulation results clearly indicate the effectiveness of our suggested approach over classical K-means for all the five investigated problems in our work. Our suggested approach overcomes the local optima problem faced by K-means with the swapping of best particles in parallel approach and delivering good solution in transitional approach.

As further work, we will like to explore how our method behaves with large data set having large dimension. Also we will like to explore the data clustering for categorical data set and mixed data set.

References

[1] I. E. Evangelou, D. G. Hadjimitsis, A. A. Lazakidou, Clayton, "Data Mining and Knowledge Discovery in Complex Image Data using Artificial Neural Networks," Workshop on Complex Reasoning an Geographical Data ,Cyprus, 2001**.**

[2] T. Lillesand and R. Keifer, *Remote Sensing and Image Interpretation*, John Wiley & Sons. 1994

[3] H. C. Andrews, *Introduction to Mathematical Techniques in Pattern Recognition,* John Wiley & Sons, New York. 1972.

[4] M. R. Rao, "Cluster Analysis and Mathematical Programming," *Journal of the American Statistical Association*, vol. 22, pp. 622-626, 1971.

[5] Kennedy, R. C. Eberhart, Y. Shi, "Swarm Intelligence," *Morgan Kaufmann*, 2002.

[6] U. Maulick and S. Bandyopadhyay, "Genetic Algorithm based Data Clustering Techniques," *Pattern Recognition*, vol-33, pp. 1455-1465, 2000.

[7] Storn R. and Price K., "Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, no.4, pp. 341–359, 1997.