AN EFFICIENT PSO-BASED CLUSTERING ALGORITHM

Chun-Wei Tsai, Ko-Wei Huang, Chu-Sing Yang
Department of Electrical Engineering, National Cheng Kung University, Tainan 70101, Taiwan

Ming-Chao Chiang
Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan

Keywords: Data clustering, Swarm intelligence, Particle swarm optimization.

Abstract: Recently, particle swarm optimization (PSO) has become one of the most popular approaches to clustering problems because it can provide a higher quality result than deterministic local search method. The problem of PSO in solving clustering problems, however, is that it is much slower than deterministic local search method. This paper presents a novel method to speed up its performance for the partitional clustering problem—based on the idea of eliminating computations that are essentially redundant during its convergence process. In addition, the multistart strategy is used to improve the quality of the end result. To evaluate the performance of the proposed method, we compare it with several state-of-the-art methods in solving the data and image clustering problems. Our simulation results indicate that the proposed method can reduce from about 60% up to 90% of the computation time of the *k*-means and PSO-based algorithms to find similar or even better results.

1 INTRODUCTION

Clustering (Xu and Wunsch, 2008) has been widely used in a variety of areas such as information retrieval, image processing, pattern recognition, just to name a few. The clustering problem refers to the process of splitting dissimilar data into disjoint clusters and grouping similar data into the same cluster based on some predefined similarity metric. The optimal partitional clustering is a partitioning that minimizes the intra-cluster distance and maximizes the inter-cluster distance. Mathematically, given a set of ℓ-dimensional patterns $X = \{x_1, x_2, \ldots, x_n\}$, the output of an optimal clustering are a partition $\Pi = \{\pi_1, \pi_2 \ldots, \pi_k\}$[1] and a set of means or centroids $C = \{c_1, c_2, \ldots, c_k\}$ such that

$$c_i = \frac{1}{|\pi_i|} \sum_{x \in \pi_i} x, \quad (1)$$

and

$$\pi_i = \{x \in X \mid d(x, c_i) \leq d(x, c_j), \forall i \neq j\}, \quad (2)$$

where d is a predefined function for measuring the similarity between patterns and means. The predefined function depends, to a large extent, on the application. For instance, for the image clustering problem, Euclidean distance is widely used. For the document clustering problem, the angle between patterns is employed. Known to be NP-hard (Kogan, 2007), the clustering algorithm usually takes a tremendous amount of time to find the solution. For this reason, many researchers (Xu and Wunsch, 2008) have focused their attention on either finding a better solution or accelerating its speed.

Recently, many partitional clustering algorithms based on population-based metaheuristic (PBM) have been proposed. Among them are the partitional clustering algorithms based on genetic algorithm (GA) (Raghavan and Birchand, 1979) and particle swarm optimization (Omran et al., 2002). An important advantage of these algorithms is that they can be employed to avoid converging to the nearest local optimum (e.g., *k*-means) from the starting position of the search (Paterlini and Krink, 2006). But in most cases, PBM takes much more computation time than single-solution-based and deterministic local search algorithms. This is one of the reasons why it is important to reduce the computation time of PBM-based clustering algorithms.

[1] That is, $X = \cup_{i=1}^{k} \pi_i$ and $\forall i \neq j, \pi_i \cap \pi_j = \emptyset$.

Tsai C., Huang K., Yang C. and Chiang M..
AN EFFICIENT PSO-BASED CLUSTERING ALGORITHM.
DOI: 10.5220/0003055301500155
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 150-155
ISBN: 978-989-8425-28-7

This paper presents an efficient algorithm called MPREPSO (Multistart and Pattern Reduction Enhanced Particle Swarm Optimization) for enhancing the speed and quality of PSO-based clustering algorithms. Then, we compare the performance of MPREPSO with five state-of-the-art clustering algorithms, with a focus on the data and image clustering problems. The remainder of the paper is organized as follows. Section 2 gives a brief introduction to the PSO algorithms for clustering problem. Section 3 provides a detailed description of the proposed algorithm. Performance evaluation of the proposed algorithm is presented in Section 4. Conclusion is given in Section 5.

2 RELATED WORK

The conventional PSO (Kennedy and Eberhart, 1995) uses the position and velocity of particles to emulate the social behavior. The position represents a trial solution of the optimization problem (e.g., clustering problem) while the velocity represents the search direction of the particle. Initially, all the particles are randomly put in the search space, and all the velocities are also randomly generated. The velocity and position of each particle at iteration $t+1$ are defined, respectively, by

$$v_i^{t+1} = \omega v_i^t + a_1\varphi_1(\mathrm{pb}_i^t - p_i^t) + a_2\varphi_2(\mathrm{gb}^t - p_i^t), \quad (3)$$

and

$$p_i^{t+1} = p_i^t + v_i^{t+1}, \quad (4)$$

where v_i^t and p_i^t represent the velocity and position of the i-th particle at iteration t; pb_i^t the personal best position of the i-th particle up to iteration t; gb^t the global best position so far; ω an inertial weight; φ_1 and φ_2 two uniformly distributed random numbers used to determine the influence of pb_i and gb; a_1 and a_2 two constants indicating, respectively, the cognitive and social learning rate.

To the best of our knowledge, the idea of the PSO-based clustering algorithm first introduced by Omran et al. (Omran et al., 2002) is to encode the k centroids as the position of a particle; that is, $p_i = (c_{i1}, c_{i2}, \ldots, c_{ik})$ where c_{ij} represents the j-th centroid encoded in the i-th particle. The fitness of each particle is defined by

$$f(p_i, M_i) = w_1\bar{d}_{\max}(p_i, M_i) + w_2[z_{\max} - d_{\min}(p_i)], \quad (5)$$

where

$$\bar{d}_{\max}(p_i, M_i) = \max_{j=1,\ldots,k}\left[\sum_{\forall x\in\pi_{ij}} \frac{d(x, c_{ij})}{|c_{ij}|}\right], \quad (6)$$

indicates the maximum mean square intra-cluster distance;

$$d_{\min}(p_i) = \min_{\forall a,b,a\neq b} d(c_{ia}, c_{ib}), \quad (7)$$

indicates the minimum distance between all the centroids. Moreover, M_i is the matrix representing the assignment of patterns to the clusters encoded in the i-th particle; $z_{\max}$ the maximum feature value in the dataset (e.g., the maximum pixel value in an image set); w_1 and w_2 the two user defined constants.

In addition to data clustering, the PSO-based clustering algorithm has been successfully applied to many other problems such as gene clustering and vector quantization. For instance, Xiao et al. (Xiao et al., 2003) integrated PSO with self organizing map for clustering gene expression data of Yeast and Rat Hepatocytes. Feng et al. (Feng et al., 2007) used the fuzzy inference method to determine which training vector will belong to which codeword and combine it with PSO to improve the quality of the end result.

3 THE PROPOSED ALGORITHM

The proposed algorithm MPREPSO is as outlined in Fig. 1. In words, assuming m is the population size of PSO, the proposed algorithm first randomly selects m subsets of patterns from the set of input patterns X. Each subset is composed of a certain percent of the patterns in X and is associated with a particle of PSO. Then, MPREPSO applies k-means to each particle and uses the result thus obtained as the initial position of the corresponding particle of PSO. The intention of this sampling method is to improve the accuracy rate of the clustering result. In addition to all the operators of PSO, which include operator to update the centroids, operator to assign patterns to all the clusters, operator to update the personal best and the global best, and operator to change the velocities and positions, MPREPSO adds to PSO three additional operators: *detection*, *compression*, and *multi-start*. The detection and compression operators take the responsibility of detecting and compressing the static patterns. The multi-start operator is added to improve the quality of the end result—by enforcing the diversity of the population of MPREPSO.

3.1 The Detection Operator

The proposed algorithm relies on two different kinds of detection operators to find computations that are essentially redundant. More precisely, the first detection operator (Tsai et al., 2007) considers patterns within a predefined radius γ to their centroid as static.

1. Randomly select m subsets of patterns, denoted s_i for $i = 1, 2, \ldots, m$, from X by **sampling**.
2. Create an initial population of m particles each of which encodes the k centroids obtained by applying k-means to all the s_i.
3. For each particle i
4. For each pattern $x \in X$
5. Calculate the distance of x to all the centroids, denoted c_{ij} for $j = 1, 2, \ldots, k$.
6. Assign x to the cluster the centroid of which is nearest to x.
7. End
8. Calculate the fitness value using Eq. (5).
9. **Detect** the set of patterns R that are static and within a predefined radius γ to their centroid.
10. **Compress** the set of patterns R into a single pattern r and remove R; that is, $X = X \cup \{r\}$ and $X = X \setminus R$.
11. End
12. Update the personal best pb_i and the global best gb using Eqs. (3) and (4).
13. Change the velocities and positions.
14. **Perform** the multi-start operator.
15. If the stop criterion is satisfied, then stop and output the best particle; otherwise, goto step 3.

Figure 1: Outline of MPREPSO for the clustering problem.

This operator uses a simple approach to determining for each cluster the top α% of patterns that can be considered as static, by using the standard deviation σ, mean μ, and confidence intervals of patterns in that cluster so that no sorting is required. This cuts the time complexity from $O(n \log n)$ down to $O(n)$. For instance, we are assuming that the distances of all the patterns in a cluster to their centroid are normally distributed. As a consequence, to find the top 16% of patterns that are close to the centroid of a cluster, the detection operator only needs to compute the average distance (μ) of the patterns to their centroid and the standard deviation σ. A pattern will be in the top 16% (i.e., $(100-68)/2\% = 16\%$) if its distance to the centroid is smaller than $\gamma = \mu - \sigma$. Note that as far as this paper is concerned, $\alpha = 16\%$ is used by the first method.

The second detection operator checks to see if a static pattern can be eliminated by counting the number of iterations that pattern stays in the same cluster. Intuitively, the higher the number of iterations a pattern remains in the same cluster, the more likely the pattern is static. How many iterations a pattern needs to stay in the same group depend on the convergence speed or the quality of the end result. If we set the number of iterations to a large value, the accuracy rate will be high, but the downside is that the computation time will increase. On the other hand, if we set the number of iterations to a small value, the result will be the other way around. Again, note that as far as this paper is concerned, two iterations in a row are used.

3.2 The Compression Operator

As the name suggests, the compression operator of MPREPSO takes care of compressing the "static" patterns of each cluster into a single pattern and ensuring that all the other operators of PSO will work on the compressed space so as to get rid of computations that are redundant. More precisely, MPREPSO first compresses all the patterns in R into a single pattern r. Then, the compression operator ensures that all the other operators will see only the pattern r instead of all the patterns in R by adding r to X and removing all the patterns in R from X. In other words, seeing only the pattern r, no redundant computations will be done for the patterns in R, including centroid update, pattern assignment, fitness value computation, particle update.

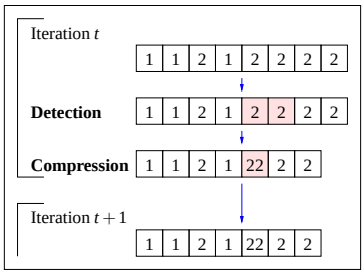

Figure 2: A simple example illustrating how the compression operator works.

As Fig. 2 shows, one of the solutions in a particular particle has eight patterns before compression. But after the detection, the proposed algorithm finds that two of the eight patterns in cluster 2 can be considered as static. The compression operator will first save away information relevant to these two patterns and then compress them into a single pattern r. After that, seven patterns (six patterns plus r) need to be computed by the other operators of PSO. As such, MPREPSO can reduce the computation time by "eliminating" on the fly patterns that are static.

3.3 The Multi-start Operator

In this paper, the multi-start method is used to improve the quality of the end result—by enforcing the diversity of PSO. The design concept of this method is to retain the structure of high performance solutions (intensification) and then use it to seek other good solutions that are similar to the current good solutions but not the same (diversification). The main purpose of this operator is similar to that proposed in (Larrañaga and Lozano, 2002; Tsai et al., 2002) but not performed once every iteration of the convergence process.

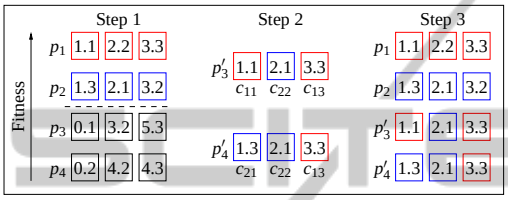

Figure 3: A simple example illustrating how the multistart operator works for MPREPSO.

As Fig. 3 shows, the multi-start operator takes three steps. Assuming that there are four particles (solutions) at the current iteration, the first step is to remove the particles the fitness of which are below the average fitness of all the particles such as p_1 and p_2. In other words, the first step is responsible for "intensifying" the good search directions, by passing the fitter solutions on to later iterations. The second step is to create new particles by a random clone method. For example, the remaining particles are $p_1 = \{c_{11}, c_{12}, c_{13}\}$ and $p_2 = \{c_{21}, c_{22}, c_{23}\}$ each of which encodes three centroids. The first centroid of the new particle can take the value of either c_{11} or c_{21}. As a result, the sub-solutions of p'_3 are c_{11}, c_{22}, and c_{13}, respectively. This step plays the role of "diversifying" the current search trajectories to avoid the proposed algorithm from falling into local optimum at early iterations. The third step is to combine the particles that have a high fitness value (p_1 and p_2) at the current iteration with the new particles p'_3 and p'_4 to generate a new population for the next iteration. Note that as far as all the experimental results given in this paper are concerned, the multi-start operator is performed once every one-tenth of the total number of iterations.

4 PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed algorithm by using it to solve the clustering problem. The empirical analysis was conducted on an IBM X3400 machine with 2.0 GHz Xeon CPU and 8GB of memory running CentOS 5.0 with Linux 2.6.18; the programs are written in Java 1.6.0_07. To evaluate the performance of MPREPSO for the PSO-based algorithms, we apply it to four PSO-based algorithms that are: PSO (Omran et al., 2005), particle swarm clustering (PSC) (Cohen and de Castro, 2006), time-varying acceleration coefficients with MPSO (TVAC) (Ratnaweera et al., 2004), and comparative PSO (CPSO) (Yang et al., 2008). We also compare the results of these algorithms with *k*-means (KM) (McQueen, 1967).

Two different kinds of datasets, dataset 1 (DS1) and dataset 2 (DS2), are used to evaluate the performance of these algorithms, as shown in Table 1. In addition, all the images in DS2 are of size 512×512 and in 8-bit grayscale. For DS1, the number of clusters of KM and PSO-based algorithms are predefined by the test problems; for DS2, the number of clusters are set to 8.

All the simulations are carried out for 30 runs.

Table 1: Datasets for benchmarks.

Dataset	Type	Name of datasets		
DS1	Data	iris	wine	breast cancer
DS2	Image	Lena	baboon	airplane
		peppers	goldhill	boots

For all the PSO-based algorithms, the population size is defaulted to 20. The number of iterations is set to 1,000. The other parameter settings are summarized in Table 2. The mutation rate of TVAC is set to 0.8. For each particle of PSO, MPREPSO uses 2% of the input patterns as the samples to create the initial solution. The maximum velocity $v_{\max}$ of PSO-based algorithms is set to 0.01 (Cohen and de Castro, 2006) for DS1 and 255 (Omran et al., 2005) for DS2. In addition, the other settings of all the PSO-based algorithms we compared in this paper follow those described in the corresponding papers. In Table 2, the settings of CPSO are the same as PSO.

To simplify the discussion of the simulation results, we will use the following conventions. Let $\beta \in \{D, T\}$ denote the quality of the clustering result ($\beta = D$) and the computation time ($\beta = T$), respectively. Also, let Δ_β denote the enhancement of

Table 2: Data settings for clustering algorithms.

Algorithms	The settings		
PSO	$\omega = 0.72$	$a_1 = 1.49$	$a_2 = 1.49$
PSC	$\omega = 0.95$	$a_1 = a_2 = [0.1, 2.05]$	$a_3 = [0.005, 1]$
TVAC	$\omega = 0.9$ to 0.4	$a_1 = 2.5$ to 0.5	$a_2 = 0.5$ to 2.5

Table 3: Enhancement of the running time of KM, PSO, PSC, TVAC, and CPSO with PR.

Data clustering (Δ_T)					
Data	KM	PSO	PSC	TVAC	CPSO
iris	-84.6	-65.1	-72.5	-79.0	-76.8
wine	-82.9	-68.4	-66.8	-82.1	-81.4
breast	-79.6	-69.1	-62.9	-82.4	-81.7
Average	-82.4	-67.5	-67.4	-81.2	-80.0
Image clustering (Δ_T)					
Data	KM	PSO	PSC	TVAC	CPSO
Lena	-76.9	-72.9	-90.2	-90.0	-74.7
baboon	-76.2	-72.3	-93.7	-89.6	-73.9
airplane	-79.1	-72.1	-93.5	-88.4	-74.0
pepper	-78.8	-74.7	-88.0	-87.8	-74.7
goldhill	-77.0	-72.0	-90.9	-87.4	-74.1
boots	-77.1	-72.2	-86.4	-88.3	-75.4
Average	-77.5	-72.7	-90.5	-88.6	-74.5

Table 4: Enhancement of the quality of KM, PSO, PSC, TVAC, and CPSO with PR.

Data clustering (Δ_{AR})					
Data	KM	PSO	PSC	TVAC	CPSO
iris	-4.9	0.1	0.8	-3.0	0.1
wine	-4.8	-0.4	-0.1	-0.1	-0.1
breast	-0.1	0.2	-0.6	-2.0	0.2
Average	-3.3	-0.04	0.01	-1.7	0.1
Image clustering (Δ_{PSNR})					
Data	KM	PSO	PSC	TVAC	CPSO
Lena	0.1	5.0	3.4	2.6	-0.5
baboon	1.4	3.2	1.3	-1.1	5.2
airplane	0.1	0.2	-1.1	0.5	-0.1
pepper	-0.9	1.3	1.0	0.5	-0.9
goldhill	0.2	0.6	0.5	0.4	-0.5
boots	3.2	6.3	-5.0	-2.1	4.0
Average	0.7	2.8	0.02	0.1	1.2

β_ϕ (new algorithm) with respect to β_ψ (original algorithm) in percentage, and it is defined by

$$\Delta_\beta = \frac{\beta_\phi - \beta_\psi}{\beta_\psi} \times 100\% \quad (8)$$

Note that for $\beta = D$, the larger the value of Δ_β, the greater the enhancement; for $\beta = T$, the smaller the value of Δ_β, the greater the enhancement. In addition, for DS1, the quality of the clustering result is measured in terms of the accuracy rate (AR) defined by

$$\mathrm{AR} = \frac{\sum_{i=1}^{n} A_i}{n}, \quad (9)$$

where A_i assumes one of the two values 0 and 1, with $A_i = 1$ representing the pattern x_i is assigned to the right cluster and $A_i = 0$ representing the pattern x_i is assigned to the wrong cluster. For DS2, the quality of the end result is measured using peak-signal-to-noise ratio (PSNR).

4.1 The Simulation Results

Our simulation contains KM, PSO, PSC, TVAC, CPSO, and MPREPSO for both the data and image clustering problems in terms of both the running time and the quality (measured, respectively, by AR and PSNR). The detection operator of MPREPSO considers a pattern as static if its distance to the centroid is no larger than $\gamma = \mu - \sigma$ and if it stays in the same group for two iterations. Our simulation results show that *k*-means (KM) is faster than the other PSO-based algorithms in most cases. However, our simulation results show further that all the PSO-based algorithms give better results than KM for most of the datasets evaluated.

Tables 3 and 4 compare the proposed algorithm MPREPSO with the other algorithms in terms of both the running time and the quality. Table 3 shows that the proposed algorithm can reduce the computation time of these clustering algorithms from 67% up to 90% on average, especially for large datasets. For example, as the results of Table 4 show, the proposed algorithm can reduce more of the computation time of PSO for DS2 than for DS1 because the data size of DS2 is larger than that of DS1. Moreover, for some datasets, the proposed method will degrade the quality of the end results, though by no more than 4% on average. For the others, the proposed method can even enhance the quality of the end result by about 0.01% up to 2.78%, especially for DS2. Our observation shows that these enhancements are due to the fact that both sampling and multi-start are used by the proposed algorithm to improve the quality of the end result. A closer look at the results shows that the proposed algorithm can reduce most of the computation time of PSO and its variants in computing fitness and updating membership of patterns. However, since each PSO-based algorithm may use different operators, the amount of time that MPREPSO can reduce is different.

5 CONCLUSIONS

This paper presents a method, based on the notion of pattern reduction, to reduce the running time of PSO-based clustering algorithms. The simulation result shows that many of the computations on the convergence process of PSO are essentially redundant and can be detected and eliminated. The simulation result shows further that the proposed algorithm can not only significantly reduce the computation time of PSO-based algorithms for clustering problems, it can also provide better results than the other algorithms

we compared in this paper. In the future, our goal is to focus on finding an even more efficient detection and multi-start method to enhance the quality.

ACKNOWLEDGEMENTS

This work was supported in part by National Science Council, Taiwan, ROC, under Contract Nos. NSC98-2811-E-006-078 and NSC98-2219-E-006-002.

REFERENCES

Cohen, S. C. and de Castro, L. N. (2006). Data clustering with particle swarms. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1792–1798.

Feng, H.-M., Chen, C.-Y., and Ye, F. (2007). Evolutionary fuzzy particle swarm optimization vector quantization learning scheme in image compression. *Expert Systems with Applications*, 32(1):213–222.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948.

Kogan, J. (2007). *Introduction to Clustering Large and High-Dimensional Data*. Cambridge University Press, New York, NY, USA.

Larrañaga, P. and Lozano, J. A., editors (2002). *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation (Genetic Algorithms and Evolutionary Computation)*. Springer, Norwell, MA, USA.

McQueen, J. B. (1967). Some methods of classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297.

Omran, M. G., Engelbrecht, A. P., and Salman, A. A. (2005). Particle swarm optimization method for image clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(3):297–321.

Omran, M. G., Salman, A. A., and Engelbrecht, A. P. (2002). Image classification using particle swarm optimization. In *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning*, pages 370–374.

Paterlini, S. and Krink, T. (2006). Differential evolution and particle swarm optimisation in partitional clustering. *Computational Statistics & Data Analysis*, 50(5):1220–1247.

Raghavan, V. and Birchand, K. (1979). A clustering strategy based on a formalism of the reproductive process in a natural system. In *Proceedings of the Second International Conference on Information Storage and Retrieval*, pages 10–22.

Ratnaweera, A., Halgamuge, S. K., and Watson, H. C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3):240–255.

Tsai, C.-F., Tsai, C.-W., and Yang, T. (2002). A modified multiple-searching method to genetic algorithms for solving traveling salesman problem. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 3.

Tsai, C.-W., Yang, C.-S., and Chiang, M.-C. (2007). A time efficient pattern reduction algorithm for *k*-means based clustering. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 504–509.

Xiao, X., Dow, E. R., Eberhart, R., Miled, Z. B., and Oppelt, R. J. (2003). Gene clustering using self-organizing maps and particle swarm optimization. In *Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, page 154.2.

Xu, R. and Wunsch, D. C. (2008). *Clustering*. Wiley, John & Sons, Inc.

Yang, C.-S., Chuang, L.-Y., Ke, C.-H., and Yang, C.-H. (2008). Comparative particle swarm optimization (cpso) for solving ptimization problems. In *Proceedings of International Conference on Research, Innovation and Vision for the Future in Computing & Communication Technologies*, pages 86–90.