PARALLEL PARTICLE SWARM OPTIMIZATION IN DATA CLUSTERING

YASIN ORTAKCI

Karabuk University, Computer Engineering Department, Karabuk, Turkey
E-mail: yasinortakci@karabuk.edu.tr

Abstract — Particle Swarm Optimization (PSO) is a heuristic and population based optimization algorithm. PSO can be used in clustering problems and dominates well known clustering algorithms such as K-Means and Fuzzy C-Means in the context of not being stuck the local optima. In this study, a parallel PSO method was presented for clustering data.The parallel PSO was tested with the iris and an artificial data set on a multiprocessor system. The results are compared to the sequential PSO in term of fitness value, computation time and clustering success. The results show that parallel PSO outclasses sequential PSO in the term of computation time in multiprocessor system. On the other hand, clustering success of parallel PSO is not less than sequential PSO. Even its clustering success is %100 for the artificial dataset, whereas S-PSO's clustering success is about %95. Besides, parallel PSO converges to the optimum result for both datasets much earlier than S-PSO.

Index Terms — PSO, parallel computing, clustering, multiprocessor.

I. INTRODUCTION

Clustering is a grouping process of unlabeled dataset without any intervention of a supervisor. The aim of clustering is decomposing a dataset with n elements to k different groups. Each decomposed cluster consists of elements that are similar between themselves and dissimilar to the elements of other clusters [1], [2]. Clustering algorithms have an extensive usage in machine learning, data mining, data analysis, marketing, pattern recognition, image segmentation and many engineering fields [3].

In this study, a centroid based clustering algorithm has been developed using Particle Swarm Optimization (PSO). PSO is a heuristic and population based optimization algorithm proposed by Russell Eberhart and James Kennedy in 1995 [4]. They developed PSO by inspiring social behavior of a swarm such as bird flocking and fish schooling while the swarm was looking for food collectively. PSO outclasses many searching algorithm in the literature since it does not need so many parameters and finds the optimum value quickly [5]. Besides, PSO can be used in clustering problems and dominates well known clustering algorithms such as K-Means [6], [7] and Fuzzy C-Means [8], [9] in the context of not being stuck the local optima.

On the other hand, in this study, parallel computing techniques were implemented to the proposed PSO based clustering algorithm for decreasing computational time on the multiprocessor systems. An algorithm must be decomposed to independently running individual parts to be able to implement parallelization on it. The main goal of a parallelization is executing the tasks concurrently on the separate processors in a multiprocessor system without no ill effects and providing a fair load balancing between the processors [10]. PSO is a suitable algorithm for parallelization since it is population based algorithm that consists of independently running particles that generally executes individual operations.

The first researches on the parallelization of PSO were done by Gies and Rahmat-Samii in 2003 on reconfigurable array design of antenna [11] and by Schutte et al. in 2004 on the biomechanical sample applications [12], respectively. Besides, Venter implemented parallel PSO on the aircraft wing aerodynamic analysis [13]. Kalivarapu et. al. presented a synchronous parallel PSO with digital pheromones to coordinate swarms withinn-dimensional design spaces [10]. Hung and Wang compared synchronous and asynchronous parallel PSO in [14].

On the other hand, PSO algorithm was used in many researches on the clustering subject. Data clustering with PSO, K-Means and C-Means were compared in [3], [15]. Facing classification problems in PSO was discussed in [16]. Image segmentation based on PSO clustering was implemented in [17]-[19]. All these clustering researches were done using sequential PSO. Parallelization of PSO will provide an opportunity of decreasing computational time and getting quicker response in clustering applications as in the other applications of parallel PSO. In this study, clustering with both sequential and parallel PSO were tried on the two different datasets and the results were compared.

II. SEQUENTIAL PARTICLE SWARM OPTIMIZATION (S-PSO)

PSO consists of a group of particles that take random initial position and velocity values in the definition space (D) and this particle group is called swarm. Each particle individually suggests a solution for the related PSO problem. PSO is an iterative method and is started to search best solution by initializing position (X) and velocity (V) with random numbers in the definition space. Fitness function value (F) is calculated to evaluate the solution quality of each

particle at each iteration. Fitness function takes the position of a particle as an input parameters and produces a return value indicates the quality of presented solution. Fitness function may be a minimization or maximization problem. Each particle keeps the best position and best fitness value that has been found so far. p_{best} represents the best position and $F_{p_{best}}$ represents the best fitness value of a particle respectively. Besides PSO keeps the best position of the swarm that has been found so far and called g_{best}. $F_{g_{best}}$ implies the best fitness value of the swarm so far. A PSO iteration is completed by updating position and velocity value of each particle. PSO iterates through these steps predefined maximum iteration number times or until satisfying stopping criteria.

In a PSO swarm that includes n particle, $X_i = (X_{i1}, X_{i2}, \ldots, X_{id})$ indicates position vector and $V_i = (V_1, V_{i2}, \ldots, V_{id})$ indicates velocity vector of i^{th} particle. $P_i = (P_{i1}, P_{i2}, \ldots, P_{id})$ keeps p_{best} position vector of the i^{th} particle. d indicates dimension number of problem space. Position and velocity vectors are updated according to following (1) and (2) along the iterations [19]-[21].

$$V_{ij}^{t+1} = wV_{ij}^{t} + c_1 r_1 \left(p_{best_{ij}} - x_{ij}\right) + c_2 r_2 \left(g_{best} - x_{ij}\right) \quad (1)$$

$$X_{ij}^{t+1} = X_{ij}^{t} + V_{ij}^{t+1} \quad (2)$$

$j = 1,2 \ldots., and i = 1,2, \ldots., n$
t: *Iteration number*;
c_1, c_2: *acceleration coefficients*;
r_1, r_2: *random numbers in* $[0-1]$;
w: *Inertia value*;

The new velocity value obtained from velocity update can be limited with $\pm V_{max}$. V_{max} value can be a constant value or it can be a dynamically changing value [5], [22]. Pseudo code of sequential PSO is given below:

```
BEGIN
FOR i=1 TO n
   INITIALIZE V_i and X_i RANDOMLY (-V_max ≤ V_i ≤ V_max , X_i ∈ D)
END FOR
FOR t=1 TO Max_Iteration
   FOR i=1 TO n
        CALCULATE F(i)
        UPDATE P_best (If F(i) < F(p_best))
                        F_Pbest = F(i), p_best = P_i)
        UPDATE g_best (If F(i) < F(g_best))
                        F_gbest = F(i), g_best = P_i)
        UPDATE V_i and X_i, REFER (1) AND (2)
   END FOR
END FOR
END
```

$F_{p_{best}}$: *The most feasible solution of the particle so far*
$F_{g_{best}}$: *The most feasible solution of the swarm so far*
$F(i)$: *Fitness value of* i^{th} *particle.*

III. PARALLEL PARTICLE SWARM OPTIMIZATION (P-PSO)

In parallel computing, data communication between the processors requires the shared memory usage or sometimes the tasks have the dependency between each other. In such cases, the power of parallel computing cannot be shown up. Even the parallelization expense is higher than sequential process in the context of computational time because of communication latency between processors [10], [12]. Therefore, data communication between processor must be at the minimum level to exploit parallel computing.

PSO is natural fit to the parallelization since it consists of individual particles executing generally independent operations. The particles can run on the different processors concurrently. Especially calculating fitness value is entirely separate operation for each particle. Besides, updating p_{best}, position and velocity of a particle is an individual operation respectively. However, calculating g_{best} has the dependency to the other particles' fitness value. Calculating g_{best} is done sequentially due to g_{best} is a shared value between all particles [23].

Parallelization of PSO can be done both synchronous and asynchronous. All the particles in the swarm complete each PSO step concurrently, then start to another PSO step in synchronous parallel PSO. For example, a particle, that completes to calculating its fitness value, must wait other particles' calculating their fitness values before starting to update the velocity and position. But however, PSO steps can be implemented by each particle independently in asynchronous parallel PSO [13]. In this study, a synchronous parallel PSO method was used and its pseudo code is given below:

```
BEGIN
PARALLEL FOR i=1 TO n
   INITIALIZE V_i and X_i RANDOMLY (-V_max ≤ V_i ≤ V_max , X_i ∈ D)
END PARALLEL FOR
FOR t=1 TO Max_Iteration
   PARALLEL FOR i=1 TO n
        CALCULATE F(i)
        UPDATE P_best (If F(i) < F(P_best))
                        F_Pbest = F(i), P_best = P_i)
   END PARALLEL FOR
   UPDATE g_best (If F(i) < F(g_best))
                        F_gbest = F(i), g_best = P_i)
   PARALLEL FOR i=1 TO n
        UPDATE V_i and X_i, REFER TO (1) AND (2)
   END PARALLEL FOR
END FOR
END
```

IV. CLUSTERING WITH PSO

$D = \{\overrightarrow{D_1}, \overrightarrow{D_2}, \ldots., \overrightarrow{D_m}\}$ indicates a dataset with m elements. Each element of dataset $(\overrightarrow{D_i})$ is a d-dimensional attribute vector.

Clustering algorithms splits the dataset to k different clusters, $C = \{C_1, C_2, \dots, C_k\}$. In clustering, each cluster must have elements that are similar between themselves and dissimilar to elements of other clusters. The generated clusters must have the following conditions[24]:

- Each cluster must include at least one element. $C_i \neq \emptyset$ ve $\forall i \in \{1,2,..k\}$
- Generated clusters must not include any mutual element. $C_i \cap C_j = \emptyset$ ve $\forall i \neq j$vei, $j \in \{1,2,..k\}$
- Each element of the dataset must belong to a cluster. $\bigcup_{i=1}^{k} C_k = P$

The similarity measurements of the element in the dataset is calculated with Euclidian distance [1]. For instance, similarity measurement between d-dimensional two samples, $\overrightarrow{D_i}$ and $\overrightarrow{D_j}$, is calculated in the (3).

$$\|\overrightarrow{D_i}, \overrightarrow{D_j}\| = \sqrt{\sum_{dim=1}^{d} (D_{i,dim} - D_{j,dim})^2} \quad (3)$$

PSO is able to handle clustering problems in such a way that each particle holds centroids of clusters and tries to find the most feasible cluster centroids for getting best clustering option. Each element in the dataset is scattered to the closest cluster by calculating the distance between the element and the cluster centroids. Each element is assigned to the closest cluster. Then, k clustering units occur with different number of element and forms a clustering scatter.
$X = \{X, X_2, \dots, X_n\}$ indicates an initial swarm with n particles. Each particle is a vector that represents the centroids of the clusters. $X_i = \{\overrightarrow{o_1}, \overrightarrow{o_2}, \dots, \overrightarrow{o_k}\}$ indicates a particle, k is the count of clusters, $\overrightarrow{o_j}$ indicates centroid of j^{th} cluster. $\overrightarrow{o_j} = \{o_{j,1}, o_{j,2}, \dots, o_{j,d}\}$ is a d-dimensional vector.

Clustering scatter presented by the particles are evaluated in a fitness function. The fitness function takes the generated cluster scatter as an input parameter and returns a numerical result to validate the correctness of the clustering scatter. In this study, the validation formula in (4) was used as a fitness function[25].

$$F = \frac{1}{m} \sum_{i=1}^{m} \|o_j, D_i^{C_j}\| \quad (4)$$

m: *Total numbers of elements in the dataset,*
$D_i^{C_j}$: *i^{th} element belongs to j^{th} cluster*
o_j: *Origin of j^{th} cluster*

After scattering each element of dataset to the closest cluster, the distance of each element to the centroid of its cluster is calculated and added to total, then the total is divide to total numbers of element in the dataset at (4). F is a minimization function.

The step of proposed P-PSO based clustering algorithm are given in the following pseudo code:

INPUTS: DATASET, $D = \{\overrightarrow{D_1}, \overrightarrow{D_2}, \dots, \overrightarrow{D_m}\}$
OUTPUS: CLUSTERS, $C = \{C_1, C_2, \dots, C_k\}$

```
BEGIN
PARALLEL FOR i=1 TO n
    FOR j=1 TO k
        INITIALIZE v_{i,k} AND o_{i,k} RANDOMLY
    END FOR
END PARALLEL FOR
FOR t=1 TO Max_Iteration
    FOR i=1 TO M
        FOR j=1 TO k
            CALCULATE ||o_j, Z_i|| REFER TO (3)
        END FOR
        if |o_j, Z_i|| IS MINIMUM DISTANCE, ASSIGN Z_i to C_j
    END FOR
    PARALLEL FOR i=1 TO N
        CALCULATE FITNESS, REFER TO (4)
        UPDATE P_best
    END PARALLEL FOR
    UPDATE g_best
    PARALLEL FOR i=1 TO N
        FOR j=1 TO k
            INITIALIZE v_{i,k} AND o_{i,k} RANDOMLY
        END FOR
    END PARALLEL FOR
END FOR
END
```

k: *THE NUMBER OF CLUSTER,*
C_j: *j^{th} CLUSTER,*
$\overrightarrow{o_{i,k}}$: *k^{th} CENTROID OF FOR PARTICLE i*

V. EXPERIMENTAL RESULTS

In this study, two different datasets were used to test the clustering performance of our proposed P-PSO and the results were compared to the S-PSO. The first dataset is iris flower data that comprises of 150 four dimensional elements in three different clusters. The second data set is an artificial dataset that comprises of 150 four dimensional elements in four different clusters. The 3D figures that shows the scatter of both iris and artificial dataset to their clusters were plotted in Fig. 1.The cluster names and the element numbers are given in Table I.

Table I - Information of Datasets

DATASET	CLUSTERS	NUMBER OF ELEMENT
Iris	Iris Setosa	50
	Virginica	50
	Versicolor	50
Artificial	Cluster 1	40
	Cluster 2	30
	Cluster 3	50
	Cluster 4	30

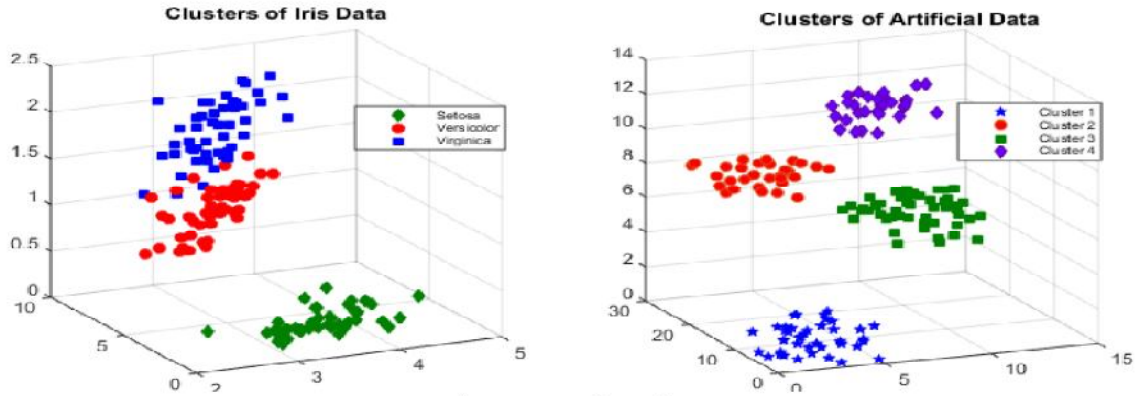

Fig. 1. The scatter of iris dataset

The applications of both S-PSO and P-PSOwere coded in Microsoft Visual Studio 2015 C#. The Task Parallel Library (TPL) was used to parallelize the S-PSO codes. Bothapplicationswere run in a netbook which has Intel® Core™ i7 5600U @ 2.60 GHz 2 cores 4 logical processorsand 8GB RAM on Windows 10 operating system. The application parameters used are acceleration coefficients,c1=1.49 and c2=1.49; inertia w=0.72; maximum iteration number, Max_Iteration=200; limit of velocity, Vmax=±5%,population size is 100.

The applications of S-PSO and P-PSO were run 15 times for both iris and artificial dataset. The fitness function results and clustering error rate (CE) are given in Table II.The clustering error rate (CE) is calculated according to (5).

$$CE = \left(\frac{wrong\ clustered\ data}{total\ data\ numbers} \times 100\right) \quad (5)$$

In Table II, it is obviously seen that computation time of P-PSO is approximately twice faster than S-PSO. They both have similar fitness values for both of the dataset. When we compare the clustering error rate, S-PSO shows a little better performance according to P-PSO for iris dataset. However, P-PSO shows excellent performance for artificial dataset and outclasses S-PSO. P-PSO find the correct clustering at all of the runs, whereas clustering error rate of S-PSO is %6.2.

The fitness function results of S-PSO and P-PSO along the 200 iterations are depicted for both iris and artificial dataset in Fig. 2. P-PSO converges to the optimum result for both datasets much earlier than S-PSO.

TableII - Results of ClusteringCode Runs

EVALUATION CRITERIAS		IRIS		ARTIFICIAL	
		S-PSO	P-PSO	S-PSO	P-PSO
FITNESS VALUES	BEST	0,0314	0,0314	0.04544	0.04544
	WORST	0,0318	0.0319	0.06055	0.04545
	MEAN	0,0315	0.0316	0.048223	0.045446
COMPUTATION TIMES (sec.)	BEST	2.61	1.15	3.16	1.36
	WORST	2.65	1.30	3.28	1.63
	MEAN	2.64	1.25	3.22	1.43
	CE(%)	4.75	5.73	6.20	0.00

CONCLUSION

In this study, P-PSO method was presented for clustering data and it was tested on the iris and artificial datasets. The results were compared to S-PSO's result in the context of fitness value, computation time and clustering error rate. The results show that P-PSO outclasses S-PSO in the term of computation time in multiprocessor system. On the other hand, clustering success of P-PSO is not less than S-PSO. Even its clustering success is %100 for the artificial dataset,whereas S-PSO's clustering success is about %95. As obviously seen in Fig. 2, the convergence of P-PSO occurs much earlier than S-PSO's convergence.

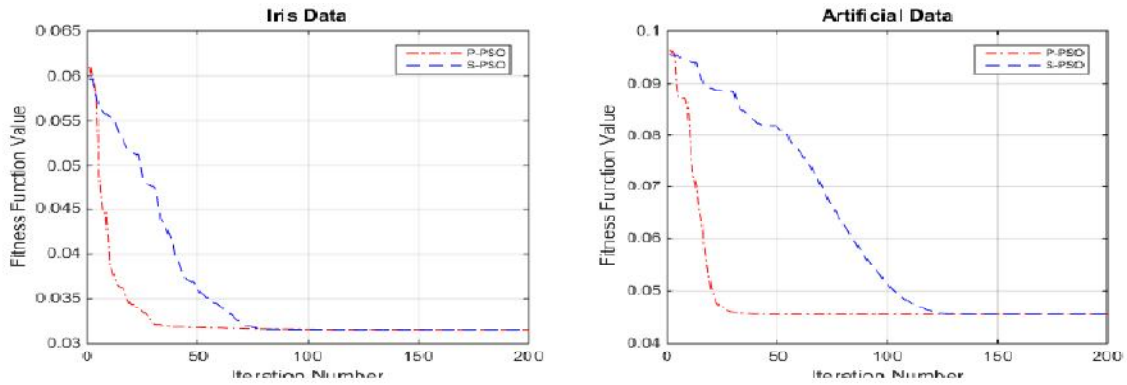

Fig. 2 Fitness values

But it can be said that, the clustering success of P-PSO in iris dataset is not high as much as it is in artificial dataset. When Fig. 1 is examined, it is seen that iris clusters, versicolor and virginica are interwoven. It is so difficult to scatter interwoven datasets to the correct clusters by using distance based similarity measurements. But however, P-PSO show good performance for the artificial dataset which has exactly discrete clusters.

As a result, P-PSO can be used in multiprocessor systems and shows good performance in terms of the computation time and the correctness of the clustering especially for the dataset that composed of discrete clusters.

REFERENCES

[1] Das, S., A. Abraham, and A. Konar, Automatic clustering using an improved differential evolution algorithm. IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans, 2008. 38(1): p. 218-237.

[2] Chen, C.-Y., H.-M. Feng, and F. Ye, Automatic particle swarm optimization clustering algorithm. International Journal of Electrical Engineering, 2006. 13(4): p. 379-387.

[3] Van der Merwe, D. and A.P. Engelbrecht. Data clustering using particle swarm optimization. in Evolutionary Computation, 2003. CEC'03. The 2003 Congress on. 2003. IEEE.

[4] James, K. and E. Russell. Particle swarm optimization. in Proceedings of 1995 IEEE International Conference on Neural Networks. 1995.

[5] Del Valle, Y., et al., Particle swarm optimization: basic concepts, variants and applications in power systems. IEEE Transactions on evolutionary computation, 2008. 12(2): p. 171-195.

[6] Likas, A., N. Vlassis, and J.J. Verbeek, The global k-means clustering algorithm. Pattern recognition, 2003. 36(2): p. 451-461.

[7] Na, S., L. Xumin, and G. Yong. Research on k-means clustering algorithm: An improved k-means clustering algorithm. in Intelligent Information Technology and Security Informatics (IITSI), 2010 Third International Symposium on. 2010. IEEE.

[8] Bezdek, J.C., R. Ehrlich, and W. Full, FCM: The fuzzy c-means clustering algorithm. Computers & Geosciences, 1984. 10(2-3): p. 191-203.

[9] Wang, H., D. Li, and Y. Chu. A New Scalability of Hybrid Fuzzy C-Means Algorithm. in Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on. 2010. IEEE.

[10] Kalivarapu, V., J.-L. Foo, and E. Winer, Synchronous parallelization of particle swarm optimization with digital pheromones. Advances in Engineering Software, 2009. 40(10): p. 975-985.

[11] Gies, D. and Y. Rahmat-Samii. Reconfigurable array design using parallel particle swarm optimization. in Antennas and Propagation Society International Symposium, 2003. IEEE. 2003. IEEE.

[12] Schutte, J.F., et al., Parallel global optimization with the particle swarm algorithm. International journal for numerical methods in engineering, 2004. 61(13): p. 2296.

[13] Venter, G. and J. Sobieszczanski-Sobieski, Parallel particle swarm optimization algorithm accelerated by asynchronous evaluations. Journal of Aerospace Computing, Information, and Communication, 2006. 3(3): p. 123-137.

[14] Hung, Y. and W. Wang, Accelerating parallel particle swarm optimization via GPU. Optimization Methods and Software, 2012. 27(1): p. 33-51.

[15] Chen, C.-Y. and F. Ye. Particle swarm optimization algorithm and its application to clustering analysis. in Networking, Sensing and Control, 2004 IEEE International Conference on. 2004. IEEE.

[16] De Falco, I., A. Della Cioppa, and E. Tarantino, Facing classification problems with particle swarm optimization. Applied Soft Computing, 2007. 7(3): p. 652-658.

[17] Li, H., H. He, and Y. Wen, Dynamic particle swarm optimization and K-means clustering algorithm for image segmentation. Optik-International Journal for Light and Electron Optics, 2015. 126(24): p. 4817-4822.

[18] Hemalatha, K., S. Ranjitha, and H. Suresh. Image segmentation based on modified centroid weight particle swarm optimization and spatial fuzzy C-means clustering algorithm. in 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT). 2015. IEEE.

[19] Omran, M., A. Salman, and A.P. Engelbrecht. Image classification using particle swarm optimization. in Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning. 2002. Singapore.

[20] Eberhart, R.C. and J. Kennedy. A new optimizer using particle swarm theory. in Proceedings of the sixth international symposium on micro machine and human science. 1995. New York, NY.

[21] Shi, Y. and R.C. Eberhart. Empirical study of particle swarm optimization. in Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. 1999. IEEE.

[22] Helwig, S., F. Neumann, and R. Wanka. Particle swarm optimization with velocity adaptation. in Adaptive and Intelligent Systems, 2009. ICAIS'09. International Conference on. 2009. IEEE.

[23] Mussi, L., F. Daolio, and S. Cagnoni, Evaluation of parallel particle swarm optimization algorithms within the CUDA™ architecture. Information Sciences, 2011. 181(20): p. 4642-4657.

[24] Das, S., A. Abraham, and A. Konar, Automatic kernel clustering with a multi-elitist particle swarm optimization algorithm. Pattern recognition letters, 2008. 29(5): p. 688-699.

[25] De Falco, I., A. Della Cioppa, and E. Tarantino. Evaluation of particle swarm optimization effectiveness in classification. in International Workshop on Fuzzy Logic and Applications. 2005. Springer.

★★★