Clustered Particle Swarm Optimization Using Self-Organizing Maps

Yongwon Park*, Harika Kilari* and Sanjeev Baskiyar*
Auburn University, Auburn, AL 36849

Abstract

Premature convergence of particles is known to be the main cause of local optima in particle swarm optimization (PSO). Such premature convergence limits the use of PSO for the multimodal function optimization problems which may have more than one optimum of a function and many local minima. In this paper, we use a novel approach called clustered PSO or CPSO, which clusters particles periodically around sample vectors (particles) using self-organizing maps. Such clustering provides sufficient diversity providing CPSO with the opportunity to explore other solutions while proactively escaping local optima. Furthermore, it enables each cluster of particles to search on the solution space concurrently for multimodal optimization problems being readily able to escape many local minima to reach the minimum. The abilities of the proposed approach in escaping local optima and finding a global solution were evaluated through simulations on the test problems used by other researchers. The results of simulations conclusively demonstrate that CPSO is highly effective in avoiding local minima for multimodal function optimization problems as well as reducing the particle population with every iteration.

Key Words: Clustering; particle swarm optimization; local minima; self-organizing maps

1 Introduction

Image synthesis of dynamic objects such as clouds, smoke, water, and fire are known to be difficult to simulate due to their fuzziness. Researchers have studied particle systems in which particles have their behavior [11] and have tried to simulate such natural dynamic objects. Many scientists have also been interested in the movement of a flock of birds or a school of fish to discover underlying rules that make possible their aggregate motion. The aggregate motion enables them to find food quickly and protects them from predators through early detection and the spread of information. Intrigued by such social behavior, Kennedy and Eberhart [4] developed the particle swarm optimization (PSO) method to apply their natural behavior to problem-solving. In PSO, particles simulate the social behavior of a flock of birds, which cooperate to find food (goal), in a way that each one remembers its personal best location ever visited called "*personal best*," and shares the local information with their neighbor to identify "*local best*" location within a group.The "*local best*" of a group is shared with other groups to identify the "*global best*" value. The local information refers to individual cognition and the global information to social interaction. Using the local and global information, a swarm of particles is able to cooperate and explore the solution space effectively to find an optimal solution. One of the advantages of PSO is the particle's prompt convergence on solutions by exploring the solution space at a fast speed like a flock of birds moving fast, but in astonishingly perfect harmony. However, the solution by early convergence may cause a local solution if done prematurely. This drawback of PSO can be attributed to its lack of ability to indicate premature convergence. The premature convergence phenomenon is commonly observed in evolutionary methods such as Genetic Algorithms (GA). In the case of a GA, it is known that high selection pressure in choosing only superior offsprings for new generations results in premature convergence. This is because the high selection pressure restricts the diversity of the new population, searching for solution space limited to local areas. Therefore, a search may get stuck in a local optimum. Riget and Vesterstrlm attributed the premature convergence of PSO to the fast information flow between particles, which causes particles to cluster early around local minima [12]. Since little diversity among the population is considered the main cause of premature convergence, many researchers [6] [18] [1] have tried to avoid premature convergence in a way that provides sufficient diversity for particles at the indication of premature convergence. The earlier methods like *k*-means clustering had an overhead of how to decide an optimum value of *k*, and also could not consider the dynamic factor in particle population update by using fixed pre-defined parameters. Later complex hierarchical and graphical clustering techniques have been devised which had low applicability due to structural limitations in data analysis. However, it remains a very difficult problem to identify the sign of premature convergence. Furthermore, using such a reactive approach may be difficult to escape local minima for optimization problems. In this research, we discuss a novel Cluster-PSO (CPSO) that we developed in 2011, using self-

*Department of Computer Science and Software Engineering Emails: yongwonpark@gmail.com, hzk0041@tigermail.auburn.edu, and baskiyar@eng.auburn.edu

organizing maps (SOM), which is known as an unsupervised learning method called "*Kohonen vector*" [5]. Alkazemi and Mohan [1], and Seo et al.[13] used clustered particles to search the solution space concurrently. However, the search may not be adaptive since the population of the group is static. On the other hand, CPSO using the SOM technique is able to make the search adaptive to the solution space by dynamically updating the population of clustered particles. The unsupervised learning method of SOM is able to cover large solution space effectively by periodically clustering particles around randomly created *Kohonen* vectors, thereby preventing local optima. The SOM helps to simulate the high dimensional topology information of particle swarm into $2D$ or $3D$ vectors by using mathematical quantification and projection.The preprocessing of raw data into clusters by SOM helps in efficient local search, which in turn helps in maintaining diversity in the optimization process.

The remainder of this paper has been organized as follows. Section 2 describes the related works, Section 3, the PSO model, and Section 4 the CPSO algorithm. In Section 5, we discuss the results and in Section 6 we make concluding remarks.

2 Related work

The PSO model simulates a swarm of particles moving in an m-dimensional solution space where a particle corresponds to a candidate solution characterized by m attributes. It is represented in the solution space by its position vector $\vec{x}_i$, and its velocity is represented by a velocity vector $\vec{v}_i$.

The velocity of the i^{th} particle of the swarm and its projected position in the d^{th} dimension are defined by the following two equations:

$$\begin{aligned}\vec{v}_{id}(t+1) = \vec{v}_{id}(t) + c_1 \cdot rand() \cdot (\vec{l}_{id} - \vec{x}_{id}(t)) \\ + c_2 \cdot rand() \cdot (\vec{g}_{id} - \vec{x}_{id}(t))\end{aligned} \tag{1}$$

$$\vec{x}_{id}(t+1) = \vec{x}_{id}(t) + \vec{v}_{id}(t+1) \tag{2}$$

where:

- n is the size of the swarm
- m is the number of dimensions in the solution space
- $i = 1, \ldots, n$
- $d = 1, \ldots, m$
- $\vec{l}_{id}$ is the local best position of particle i on the d^{th} dimension
- $\vec{g}_i$ is the global best position of particle i in the d^{th} dimension
- c_1 is the learning rate of particles for individual cognition
- c_2 is the learning rate of particles for social interaction
- $rand()$ is the random function with the output in the range $(0 \ldots 1)$

Shi and Eberhart [16] introduced a parameter *inertia weight* ω into the basic PSO:

$$\begin{aligned}\vec{v}_{id}(t+1) = \omega \cdot \vec{v}_{id}(t) + c_1 \cdot rand() \cdot (\vec{l}_{id} - \vec{x}_{id}(t+1)) \\ + c_2 \cdot rand() \cdot (\vec{g}_{id} - \vec{x}_{id}(t+1))\end{aligned} \tag{3}$$

where ω weights the magnitude of the old velocity $\vec{v}_{id}(t)$. They found the range of $(0.9 \ldots 1.2)$ a good area to choose ω from.

Many researchers have tackled the premature convergence problem in PSO. They tried to overcome it in a reactive way that provides diversity to particles at the indication of premature convergence to escape a local optimum. Krink and Riget [6] provided diversity for particles upon indication of a collision. The indication of the collision was determined based on the distance between particles, and subsequently, diversity was provided in a way that particles bounce away randomly or make a U-turn by increasing their velocity to collide against the boundary of the solution space. The tailored PSO outperformed the basic PSO for several benchmark functions. However, the reactive method may not be sufficient for complex optimization problems. Once converged at a local optimum, clustered on local best by their nature, particles would struggle to escape the local optimum without substantial diversity. On the other hand, CPSO explores solution space by explicitly clustering particles around randomly created sample vectors, thus being able to escape local optima for optimization problems. Wei, Guangbin and Dong [19] presented Elite Particle Swarm with Mutation (EPSM). EPSM tried to take advantage of best fit particles to avoid wasting time visiting the solution space with poor fitness values. To do this, particles with poor fitness were substituted by *elite particles* with better fitness. But such elitism decreases the diversity of particles. To provide diversity EPSM employed a mutation operator so that the global best particle may be mutated to generate a new particle. EPSM outperformed the Standard Particle Swarm Optimization [16] with respect to the quality of the solution. In contrast to the elitism, Wang and Qiu [18] tried to give opportunities to inferior particles to search solution space. Their approach was motivated by the observation that a search process is very likely to be dominated by several super particles, which is often not good in the long term. In order to alleviate the dominance by super particles, the selection probability of a particle was set inversely proportional to its original fitness. Next, a particle is selected in the roulette wheel manner to explore the solution space. Such a procedure is expected to mitigate the high selection pressure by super particles. Their approach outperformed other known algorithms in terms of solution quality but took additional computational time for the fitness scaling and roulette selecting process. Veeramachaneni and Osadciw [17] claimed that particles by nature oscillate between local optima and a global optimum, wasting time moving in the same direction to converge at a global optimum. Therefore, they made particles attract toward the best positions visited by their neighbors. In other words, particles are influenced by successful neighbors to explore the solution space. This algorithm was further improved by concurrent PSO implementation [2] in which two particle groups worked concurrently, with each group tracing particles independently and sharing the information about the best particle. Similarly, the Multi-Phase Particle Swarm Optimization algorithm (MPSO) employed multiple groups of particles, each changing a search direction in every phase to increase population diversity [1]. Seo et al. [13] presented multi-grouped particle swarm optimization (MGPSO) algorithm in which each group searches its own best solution independently. They prevented each group from interfering with other groups by regulating the radius of each group. Both MPSO and MGPSO provide concurrent search through clusters of particles. However, since early grouped particles search the solution space throughout the search, it may not be adaptive in a pathological environment. On the other hand, CPSO can be more adaptive to the varying environments by periodically updating the population of a group randomly. Sha and Yang [14] proposed APSO K-means clustering for speaker recognition where ant colony algorithm and PSO algorithm were combined with K-means clustering. In K-means clustering the number of groups is fixed, whilst the number of groups in CPSO is updated dynamically throughout the search of the solution space. Ratnaweera, Watson and Saman [10] used random dimension and time-varying coefficients to compute particle velocities during mutation.This technique has a drawback of loss of valuable data in some dimensions. It also has an overhead of data re-computation from scratch in case of any malfunction due to the hierarchical structure. The other major concern is outdated data, which results in diversity loss. To handle it, Li and Yang [7] devised a new dynamic optimization technique. It conducts a detailed search by dividing search space into sub-swarms. But, it faces the "*two-step forward, one-step backward*" phenomenon, as weakness in one dimension affects the overall fitness of a particle. Another dynamic technique proposed by Daniel and Xiaodong [9], Species-based Particle Swarm Optimization is effective in dealing with multimodal optimization functions in both static and dynamic environments. A networked structured PSO, called NS-PSO [8] has been proposed in which adjacent particles are connected in the neighborhood of a topological space and share the information of their best positions. These connections have been used to enhance the local search and increase diversification.

3 The CPSO Model

The CPSO is a modified version of PSO with an additional process of clustering particles. In the CPSO model, particles are periodically clustered around sample vectors using SOM to provide particles with enough diversity to prevent them from prematurely converging to local optima. The CPSO procedure has been described in Procedure 1. In Eq. 4, $\vec{V_p}(t+1)$ is the new velocity of particles, $\alpha(t)$ controls the learning rate where t is the generation number of particles and $\Phi(p,t)$ is the neighborhood function which determines the degree of the neighborhood between BMP and particle p. We took a Gaussian function as the neighborhood function for particles which denotes the lateral particle interaction and the degree of excitation of the particle. The Gaussian function which returns values between 0 and 1 is a commonly used simple model for simulating a large number of random values. Gaussian function for particles returns a value close to 1 if the particle is close to BMP (neighbors of BMP). The neighborhood function is defined [15] as:

$$\Phi(p,t) = \exp\left(-\frac{(p-b)^2}{2\alpha(t)^2}\right) \quad (6)$$

where:

- p is the current particle
- b is the best matching particle (BMP)
- t is the Time/Generation
- $\alpha(t) \in (0\dots1)$ is the learning rate

Diversity is required during early phases of optimization when local optima are computed, but as generations exhaust, the learning rate should decrease to allow convergence at global optima. Initially, the learning rate will be close to 1 and gradually it will decrease. The number of neighbors is reduced as the generation number grows. From Step 1 through Step 4, the learning process chooses the best particle from each sample vector based on fitness value and clusters the particles around the vectors by updating the positions of particles in the neighborhood. This process gives a chance to explore a new possible solution space that may contain an optimal or near-optimal solution. In Step 5, each cluster of particles is evolved by recomputing the sample vectors based on the updated particle distribution. These processes of clustering and evolving particles are iterated until the generation number is exhausted or a stopping condition that identifies no more improvements in fitness on objective functions is met. Figure 1 shows particles (clear circles) moving toward three randomly generated sample vectors (dark circles) to form clusters.

Procedure 1: Procedure CPSO

Step 1. Initialize particles
Step 2. Randomly create sample particles s in the solution space, with velocity $\vec{V_s}(t)$, where $1 \leq s \leq k$, k being the maximum number of sample vectors
Step 3. Traverse each particle p, $0 \leq p < n$, where n is the swarm size, and find the best matching particle (BMP) using the fitness value.
Step 4. Update the velocity of particles in the neighborhood of BMP by drawing them closer to sample vectors using the following formulas:

$$\vec{V_p}(t+1) = \vec{V_p}(t) + \Phi(p,t)\alpha(t)(\vec{V_s}(t) - \vec{V_p}(t)) \quad (4)$$

$$\vec{x_p}(t+1) = \vec{x_p}(t) + \vec{v_p}(t+1) \quad (5)$$

Step 5. Evolve particles using PSO.
Step 6. Go to Step 2, if $t <$ MaxGeneration and $F > \theta$ (where F is the gross increment in particle fitness on objective optimization functions, and $\theta \simeq 0$ is a very small value)

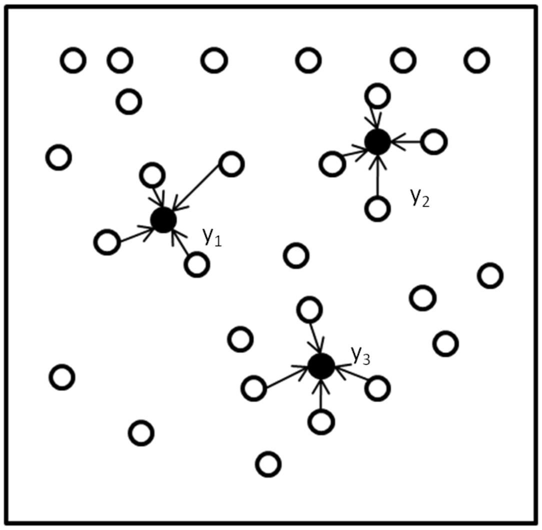

Figure 1: Clustering particles using self-organizing maps.

4 Simulations and Results

To run the tests, we implemented PSO and CPSO using the Java programming language. It is very cumbersome to conduct simulations on high-dimensional functions due to large computational overhead. For instance, if we choose a mixed range of dimension functions, it is very difficult to compare their results. The SOM uses vector projection to convert complex particle swarm topology into low-dimensional vectors. So it is economical to run simulations on a set of low-dimensional functions with different cluster sizes and generations as input. In the simulations, PSO and CPSO were run for four well-known objective functions namely DeJong's F2, Schaffer F6, Rastrigin, and Griewank. These are the same functions that were used by Kennedy [3]. The optimization performances of PSO and CPSO were compared. The corresponding objective functions have been described as follows:

$$f(x,y) = 100(x^2 - y)^2 + (1-x)^2 \tag{7}$$

where $-2.048 < x, y < 2.048$

$$f(x,y) = 0.5 + \frac{(sin^2(\sqrt{x^2 - y^2} - 0.5)}{(1 + 0.001(x^2 + y^2)^2} \tag{8}$$

where $-100 \leq x, y \leq 100$

$$f(x) = 100 + \sum_{i=1}^{10} x_i^2 - 10cos(2\pi x_i) \tag{9}$$

$$f(x) = \sum_{i=1}^{100} \frac{x_i^2}{4000} - \prod_{i=1}^{100} cos(\frac{x_i}{\sqrt{i}}) + 1 \tag{10}$$

where $-600 \leq x_i \leq 600$

De Jong's F2 function, represented by Eq. (7), is a two-dimensional function with a deep valley with the shape of a parabola. The Schaffer F6 function represented by Eq. (8) is known to be very difficult to optimize, having infinite local minima and one global minimum at $(x, y) = (0, 0)$. Rastrigin represented by Eq. (9) and Griewank represented by Eq. (10) are multimodal functions that have many local minima. Figures. 2 and 3 show the Rastrigin's function and the Griewank's function, respectively, which have many local minima shown by the "valleys." Both have the global minimum at $(0, 0, \ldots, 0)$. Figures. 4-7 show minimum fitness values found by particles exploring the solution space under the objective functions described above. Figure 4 shows the results of PSO and CPSO on the F2 function. For the F2 function, both CPSO and PSO perform well, early finding a minimum. Both converge early to a minimum, but CPSO continues to search for a better solution (which in this case does not exist). In Figure 5, it is shown that PSO prematurely converges to a solution, whereas CPSO escapes several local optima to reach global optima. Again, Figure 6 shows that for the Rastrigin's optimization problem (9), CPSO enables particles to find a global solution, oscillating between local minima and global minimum, whereas

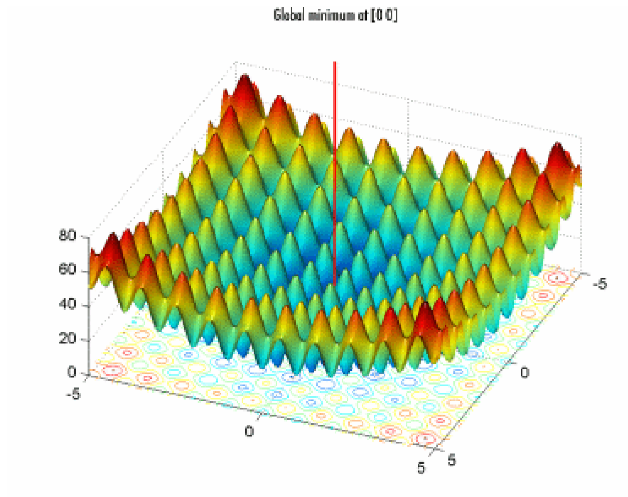

Figure 2: Rastrigin's function

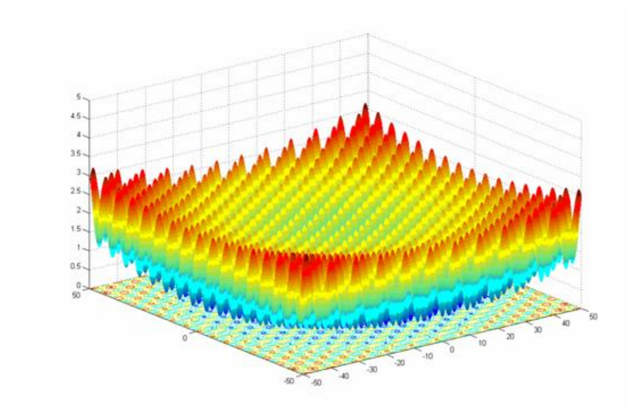

Figure 3: Griewank function

particles of PSO converge at a local minimum. Finally, Figure 7 shows CPSO outperforms PSO dramatically for a very complex multimodal optimization problem. PSO converges early at local minima, whereas CPSO quickly finds a global minimum. The results clearly demonstrate that our approach is very effective for highly complex multimodal optimization problems.

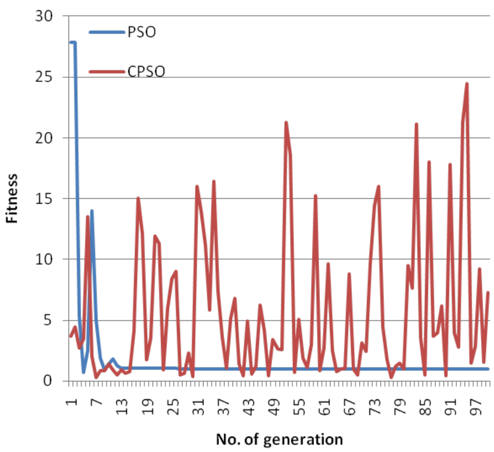

Figure 4: CPSO vs. PSO for De Jong's F2

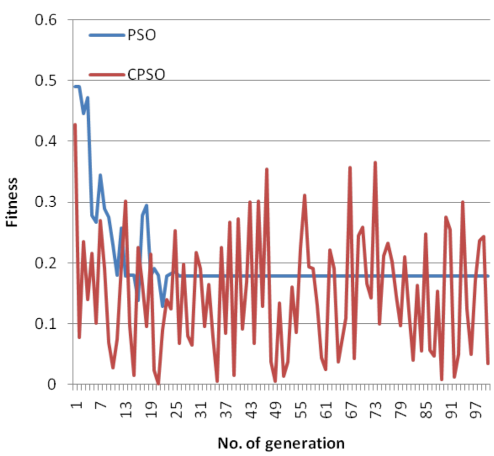

Figure 5: CPSO vs. PSO for Schaffer F6

5 Conclusions

We addressed the problem of premature convergence observed in PSO. In this research, we focused on providing enough diversity for particles to escape local minima. CPSO explicitly clusters particles around sample vectors to enable particles to escape local minima, and explore new possible solution space which may contain better solutions. Simulation results show that CPSO outperforms PSO significantly for complex optimization problems and avoids local minima yielding global solutions. Although CPSO makes an extensive

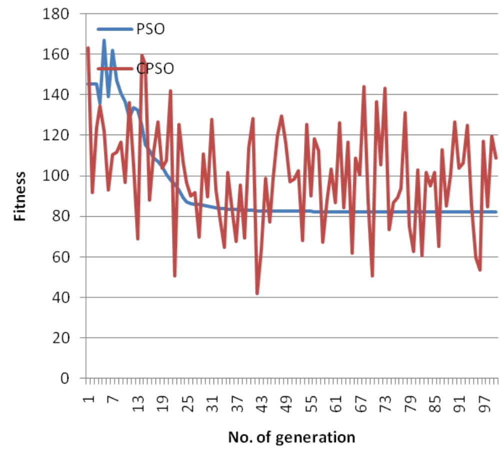

Figure 6: CPSO vs. PSO for Rastrigin F1

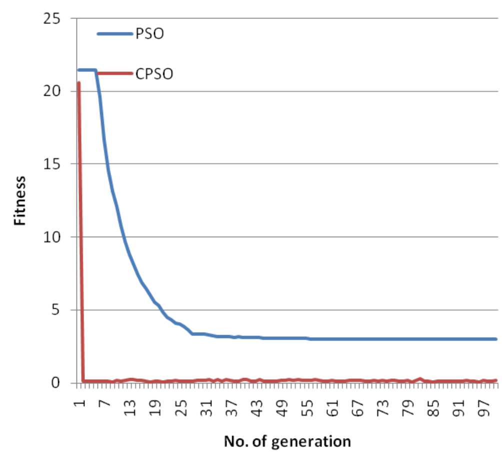

Figure 7: CPSO vs. PSO for Griewank

search within the solution space, it limits the search time by limiting the particles which explore the solution space using self-organizing maps. The research strongly suggests that CPSO is very effective for complex multimodal problems. In future work, we will study finding early signs of premature convergence for preventing such premature convergence.

Acknowledgments

We wish to thank the anonymous reviewers for their feedback which made the paper much better.

References

[1] B. Al-kazemi and C. K. Mohan, "Training Feedforward Neural Networks using Multi-phase Particle Swarm Optimization," *Proceedings of the* 9^{th} *International Conference on Neural Information Processing*, 5:2615-2619, Nov. 2002.

[2] S. Baskar and P.N. Suganthan, “A Novel Concurrent Particle Swarm Optimization,” *Proceedings of the Congress on Evolutionary Computation*, 1:792-796, June 2004.

[3] J. Kennedy, “Stereotyping: Improving Particle Swarm Performance With Cluster Analysis,” *Proceedings of the 2000 Congress on Evolutionary Computation*, 2:1507-1512, 2000.

[4] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” *Proceedings of IEEE International Conference on Neural Networks*, 4:1942-1948, Nov-Dec 1995.

[5] T. Kohonen, “The Self-Organizing Map,” *Neurocomputing*, 2(2):1-6, Nov. 1993.

[6] T. Krink, J.S. Vesterstrom and J. Riget, “Particle swarm Optimization with spatial particle extension,” *Proceedings of the Congress on Evolutionary Computation*, 2:1474-1479, 2002.

[7] C. Li and S. Yang, “A Clustering Particle Swarm Optimizer For Dynamic Optimization,” *IEEE Congress on Evolutionary Computation*, pp. 439-446, May 2009.

[8] H. Matshushita, Y. Nishio and C.K. Tse, “Network-Structured Particle Swarm Optimizer that Considers Neighborhood Distances and Behaviors,” *Journal of Signal Processing*, 18(6):291-302, Nov. 2014.

[9] D. Parrott and X. Li, “Locating And Tracking Multiple Dynamic Optima By A Particle Swarm Model Using Speciation,” *IEEE Transactions on Evolutionary Computation*, 10(4):440-458, Aug. 2006.

[10] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, “Self-Organizing Hierarchical Particle Swarm Optimizer With Time-Varying Acceleration Coefficients,” *IEEE Transactions on Evolutionary Computation*, 8(3):240-255, June 2004.

[11] W. T. Reeves, “Particle Systems–a Technique for Modeling a Class of Fuzzy Objects,” *ACM Trans. Graph.*, 2(2):91-108, 1983.

[12] J. Riget and J.S. Vesterstrom, *A Diversity-Guided Particle Swarm Optimizer - the ARPSO*, Tech Report - Dept. of Computer Science, University of Aarhus, Denmark, 2002.

[13] J.-H. Seo, C.-H. Im, C.-G. Heo, J.-K. Kim, H.-K. Jung and C.-G. Lee, “Multimodal Function Optimization Based On Particle Swarm Optimization,” *IEEE Transactions on Magnetics*, 42(4):1095-1098, 2006.

[14] M. Sha and H. Yang, “Speaker Recognition Based on APSO-K-means Clustering Algorithm,” *IEEE International Conference on Artificial Intelligence and Computational Intelligence*, 2:440-444, Nov. 2009.

[15] A. Sharma and C. W. Omlin, “Performance Comparison of Particle Swarm Optimization with Traditional Clustering Algorithms used in Self-Organizing Map,” *World Academy of Science, Engineering and Technology*, 3(3):718-729, Mar. 2009.

[16] Y. Shi and R. Eberhart, “A Modified Particle Swarm Optimizer,” *Proceedings of IEEE International Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pp. 69-73, May 1998.

[17] K. Veeramachaneni, T. Peram, C.K. Mohan and L. A. Osadciw, “Optimization using Particle Swarms with Near Neighbor Interactions,” *Lecture Notes in Computer Science - from Proceedings of the Genetic and Evolutionary Computation Conference*, 2723:110-121, 2003.

[18] F. Wang and Y. Qiu, “A Modified Particle Swarm Optimizer With Roulette Selection Operator,” *Proceedings of IEEE International Conference on Natural Language Processing and Knowledge Engineering*, pp. 765-768, Oct. 2005.

[19] J. Wei, L. Guangbin and L. Dong, “Elite Particle Swarm Optimization with Mutation,” *Asia Simulation Conference - Proceedings of the 7th IEEE International Conference on System Simulation and Scientific Computing*, pp. 800-803, Oct. 2008.

Yong-Won Park received the bachelor’s degree in Computer Science from Kwangwoon University, Seoul, Korea and the master’s and Ph.D. degrees in Computer Science and Software Engineering from Auburn University. He is a senior researcher in Samsung S1, Seoul. His research interests are in machine learning and data mining.

Harika Kilari received the B.Tech. (Honours) degree in Computer Software Engineering from IIIT, Hyderabad and the MS degree in Computer Science from Auburn University. She has worked as a Systems Software Engineer in NVIDIA, India and is currently a Senior Camera Engineer in Qualcomm, San Diego, CA.

Sanjeev Baskiyar is a Full Professor in the Department of Computer Science and Software Engineering at Auburn University, Auburn, Alabama. He received the Ph.D. and MSEE degrees from the University of Minnesota, Minneapolis, and the B.E. degree in Electronics and Communications from the Indian Institute of Science, Bangalore. His current research interests are in the areas of computer architecture, edge computing, optimization, scheduling, and real-time and embedded computing. In addition to academic appointments, he also worked in the hardware and software industry.