International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:10, No:3, 2016

Optimized Algorithm for Particle Swarm Optimization

Fuzhang Zhao

Abstract—Particle swarm optimization (PSO) is becoming one of the most important swarm intelligent paradigms for solving global optimization problems. Although some progress has been made to improve PSO algorithms over the last two decades, additional work is still needed to balance parameters to achieve better numerical properties of accuracy, efficiency, and stability. In the optimal PSO algorithm, the optimal weightings of $(\sqrt{5}-1)/2$ and $(3-\sqrt{5})/2$ are used for the cognitive factor and the social factor, respectively. By the same token, the same optimal weightings have been applied for intensification searches and diversification searches, respectively. Perturbation and constriction effects are optimally balanced. Simulations of the de Jong, the Rosenbrock, and the Griewank functions show that the optimal PSO algorithm indeed achieves better numerical properties and outperforms the canonical PSO algorithm.

Keywords—Diversification search, intensification search, optimal weighting, particle swarm optimization.

I. Introduction

PARTICLE SWARM OPTIMIZATION (PSO), originally proposed by Kennedy and Eberhart [1], [2] in 1995, is becoming one of the most important swarm intelligence paradigms [3]. PSO adopted the cognitive and social behaviors of a swarm of birds looking for food to intelligently conduct searches for global optimization.

Most global optimization problems can be converted to global minimization problems. The minimization problem considered herein is to find a global minimizer, $\mathbf{X}^*$, in a bounded D-dimensional searching space, S, which is a nonempty compact subset of $\mathcal{R}^D$, to minimize the real valued objective function, $f: S \to \mathcal{R}$ such that

$$f(\mathbf{X}^*) \le f(\mathbf{X}), \quad \forall \mathbf{X} \in S \subset \mathcal{R}^D, \tag{1}$$

in which

$$S = \prod_{d=1}^{D} [a_d, b_d], \tag{2}$$

where a_d and b_d stands for the lower and upper constant limits of a searching space in d-dimension, respectively.

In searching a D-dimensional space for problems defined in (1) and (2) by PSO, particle i of a swarm can be described by a position vector, $\mathbf{X}_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^T$ and a velocity vector $\mathbf{V}_i = (v_{i1}, v_{i2}, \cdots, v_{iD})^T$. The cognitive behavior of a swarm is transformed into tracking the previously visited positions of particles with the smallest values of an objective function by $\mathbf{P}_i = (p_{i1}, p_{i2}, \cdots, p_{iD})^T$. The social behavior of a swarm is interpreted by defining g as the index of the best particle in the swarm and by choosing the global best particle with the smallest value of the objective function among the individually visited best positions. For commonly used PSO algorithms, velocities and positions of a swarm are updated by the following two equations

$$v_{id}^{n+1} = w v_{id}^n + c_1 r_1^n (p_{id}^n - x_{id}^n) + c_2 r_2^n (p_{gd}^n - x_{id}^n), \tag{3}$$

$$x_{id}^{n+1} = x_{id}^n + v_{id}^{n+1}, \tag{4}$$

where the inertia weight, w, was proposed by Shi and Eberhart [4], the dimensional index, $d = 1, 2, \cdots, D$, the particle index, $i = 1, 2, \cdots, M$, the iteration number, $n = 1, 2, \cdots, N$, c_1 and c_2 are positive acceleration constants, and r_1 and r_2 are uniformly distributed random numbers within [0, 1]. In the diversification search, the weighted inertia velocity looks for new solutions and locates regions with potentially the best solutions. In the intensification search, the remaining two positional difference terms perturbed with random numbers explores the previous solutions and finds the best solution of a given region. Similarly, to use the inertia weight to control the maximum velocity of a swarm, a constriction factor has been introduced in analyzing the PSO convergence behavior by Clerc and Kennedy [5]. The velocities are iterated by

$$v_{id}^{n+1} = \chi \left[v_{id}^n + c_1 r_1^n (p_{id}^n - x_{id}^n) + c_2 r_2^n (p_{gd}^n - x_{id}^n) \right], \tag{5}$$

where

$$\chi = \frac{2}{\phi - 2 + \sqrt{\phi^2 - 4\phi}}, \quad c_1 + c_2 = \phi > 4.0. \tag{6}$$

When typical values of $c_1 = c_2 = 2.05$ are used, the constriction constant has the value of $\chi = 0.72984$. This is the canonical PSO algorithm [6].

Although some progress has been made to improve PSO algorithms and to expand their applications over the last two decades [7]-[10], additional work is still needed to balance parameters to achieve better numerical properties of accuracy, efficiency, and stability for PSO algorithms. An optimal PSO algorithm will be introduced in detail in the following section.

II. Optimal PSO Algorithm

In order to develop the optimal PSO algorithm with better numerical properties including accuracy, efficiency, and stability, optimizations are applied between the cognitive and social components, between the perturbation and constriction factors, and between the intensification and diversification searches.

First, the intensification search includes a cognitive component and a social component weighted by the constants,

F. Zhao conducts research free-time at 266 Robinson Drive, Tustin, CA 92782 USA (e-mail: fuzhangzhao@yahoo.com).

c_1 and c_2, respectively. When a c_1 greater than c_2 is used, each particle approaches its previous individual best position. When a c_2 greater than c_1 is used, each particle moves closer to the previous global best position. The cognitive constant c_1 and the social constant c_2 need to be optimally balanced. For an optimal balance, the golden ratio is applied to the ratio of a cognitive constant to a social constant. Thus, we have the ratio of $c_1/c_2 = (1+\sqrt{5})/2$.

Second, let us define the summation, $\phi = c_1 + c_2$, as a perturbation constant. When the perturbation constant, ϕ, increases, the depth of an intensification search increases but the constriction constant, χ, decreases based on (6), resulting in more stability but narrower diversification searches and vise versa. Equation (6) can be rearranged as

$$\chi = \frac{1}{1 + (\phi - 4 + \sqrt{\phi^2 - 4\phi})/2}. \tag{7}$$

Balance between the perturbation constant and the constriction constant can be achieved by equating the two dimensionless groups of χ and $(\phi - 4 + \sqrt{\phi^2 - 4\phi})/2$. Solving gives the optimal perturbation constant of $\phi_o = 2 + \sqrt{5}$ and the optimal constriction constant of $\chi_o = (\sqrt{5} - 1)/2$.

Third, in the diversification search, if the velocity is weighted low, the mobility of particles will be constrained and the search for potential new regions will be limited but if the velocity is weighted high, the oscillation of particles will be magnified and the efficiency of the PSO algorithm will be reduced. In the intensification search, perturbation effects due to random numbers will increase with large constants c_1 and c_2. The diversification term is the accumulation of previous intensification terms. As such, the diversification search will update and renew the intensification search. When constants c_1 and c_2, and the inertia velocity are too large, however, the stability of the PSO algorithm will deteriorate, resulting in divergent searches. For an optimal balance, the diversification search is weighted with $(1-\chi_o)$ and the intensification search is weighted by χ_o.

Combining the three aspects of optimizations, with $c_{o1} = (\sqrt{5} - 1)/2$ and $c_{o2} = (3 - \sqrt{5})/2$, the velocity recursion equation, along with the position recursion equation (4), establishes the optimal PSO algorithm

$$\begin{aligned} v_{id}^{n+1} = (1 - \chi_o) v_{id}^n + \chi_o c_{o1} \phi_o r_{1id}^n \left(p_{id}^n - x_{id}^n\right) \\ + \chi_o c_{o2} \phi_o r_{2id}^n \left(p_{gd}^n - x_{id}^n\right), \end{aligned} \tag{8}$$

where the resulting optimal parameters are $1 - \chi_o = (3 - \sqrt{5})/2$, $\chi_o c_{o1} \phi_o = (1+\sqrt{5})/2$, $\chi_o c_{o2} \phi_o = 1$, and the iteration process will not stop until the $Euclidean$ norms, η, of position changes between two consecutive iterations per a swarm size, M, satisfies a designated allowable tolerance, η_a, by

$$\eta = \frac{\|\mathbf{X}^{n+1} - \mathbf{X}^n\|}{M} = \frac{1}{M} \sqrt{\sum_{i=1}^{M} \sum_{d=1}^{D} (x_{id}^{n+1} - x_{id}^n)^2} \le \eta_a, \tag{9}$$

or a selected maximum iteration number, N, is reached. The $Euclidean$ distance averaged by a swarm size measures swarm convergent behaviors. In addition, one of the most important values to be monitored is best fitness, in which a minimized value of an objective function is evaluated at the global best location during iterations.

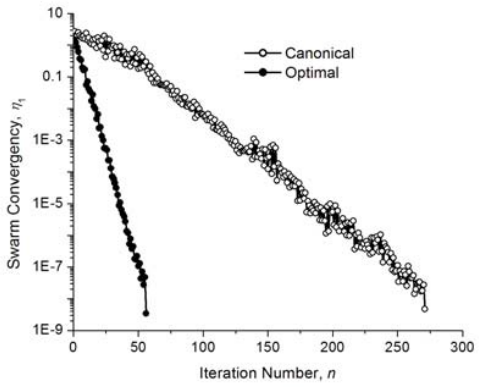

Fig. 1 Comparisons of swarm convergency between canonical and optimal PSO algorithms for the de Jong function

III. Validation of Algorithms

To validate the optimal PSO algorithm, three benchmark functions have been selected. They are the de Jong, the Rosenbrock, and the Griewank functions. Initial positions of particles are all generated within selected search domains by the uniform probability distribution function (PDF). The precision of decimal places, total iteration numbers, swarm convergent behaviors, and best fitness are collected, compared and discussed in the following section.

A. The de Jong Function

The de Jong function, $f_1(\mathbf{x})$, is given by

$$f_1(\mathbf{x}) = \sum_{d=1}^{D} x_d^2 \quad \forall \mathbf{x} \in \mathcal{R}^D, \tag{10}$$

where the global minimum is $f_1(\mathbf{x}^*) = 0$ at the minimizer of $\mathbf{x}^* = (0, \cdots, 0)^T$.

The de Jong function is the simplest test function. It is continuous, convex, and unimodal. The optimal and canonical PSO algorithms with the swarm size of $M = 6 \times 6 = 36$ and the allowable swarm convergent tolerance of $\eta_a = 10^{-8}$ have been used to search within the two-dimensional space of $[-20, 20]^2$ for the only minimum of the de Jong function.

Simulations of both the canonical and the optimal PSO algorithms have been conducted at least 10 times each. The canonical PSO algorithm yields an average precision of 12 decimal places through an average of 272 iterations while the optimal PSO algorithm produces an average precision of 9 decimal places through an average of 57 iterations. Typical simulation results for swarm convergency and best fitness are shown in Figs. 1 and 2, respectively. The canonical PSO algorithm obtains a precision of 12 decimal places through 271 iterations while the optimal algorithm produces a precision of 9 decimal places through a total of 56 iterations. When the ratio of the precision of decimal places to the total number

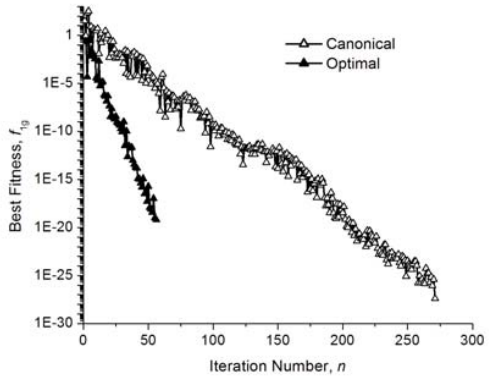

Fig. 2 Comparisons of best fitness between canonical and optimal PSO algorithms for the de Jong function

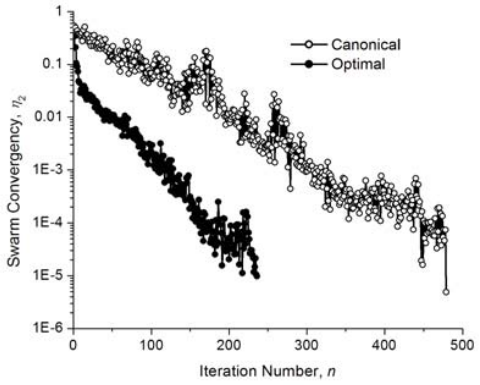

Fig. 3 Comparisons of swarm convergency between canonical and optimal PSO algorithms for the Rosenbrock function

of iterations is used to evaluate an algorithm, the canonical algorithm has a value of 0.04525 while the optimal algorithm possesses a value of 0.16056. Thus, the optimal PSO algorithm is almost 4 times as efficient and accurate as the canonical PSO algorithm.

B. The Rosenbrock Function

The Rosenbrock function, $f_2(\mathbf{x})$, is given by

$$f_2(\mathbf{x}) = \sum_{d=1}^{D-1} \left[100(x_d^2 - x_{d+1})^2 + (1 - x_d)^2\right] \quad \forall \mathbf{x} \in \mathcal{R}^D, \tag{11}$$

where the global minimum is $f_2(\mathbf{x}^*) = 0$ at the minimizer of $\mathbf{x}^* = (1, \cdots, 1)^T$.

The Rosenbrock valley is a classic optimization problem, also known as the Banana function. The global optimum is inside a long, narrow, and parabolic shaped flat valley. To find the valley is trivial, convergence to the global optimum, however, is difficult and hence this problem has been repeatedly used in assessing the performance of optimization algorithms.

The canonical and optimal PSO algorithms with the swarm size of $M = 8 \times 8 = 64$ and the allowable swarm convergent tolerance of $\eta_a = 10^{-5}$ have been used to search within the two-dimensional space of $[-3, 3]^2$ for the global minimum of the Rosenbrock function.

Simulations of both the canonical and the optimal PSO algorithms have been conducted at least 10 times each. The canonical PSO algorithm generates an average precision of 10 decimal places through an average of 478 iterations while the optimal PSO algorithm produces an average precision of 10 decimal places through only 228 iterations. Representative simulation results for swarm convergency and best fitness are shown in Figs. 3 and 4, respectively. The canonical PSO algorithm obtains a precision of 10 decimal places through 479 iterations while the optimal algorithm produces the same precision of 10 decimal places through 236 iterations. Thus, the optimal PSO algorithm is twice as efficient as the canonical PSO algorithm.

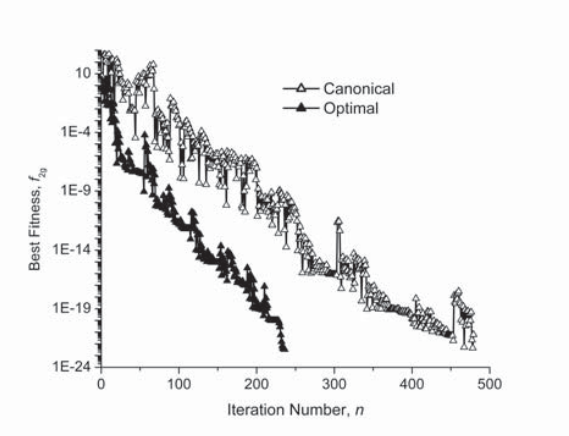

Fig. 4 Comparisons of best fitness between canonical and optimal PSO algorithms for the Rosenbrock function

C. The Griewank Function

The Griewank function, $f_3(\mathbf{x})$, is given by

$$f_3(\mathbf{x}) = \frac{1}{4000} \sum_{d=1}^{D} x_d^2 - \prod_{d=1}^{D} \cos\left(\frac{x_d}{\sqrt{d}}\right) + 1 \quad \forall \mathbf{x} \in \mathcal{R}^D, \tag{12}$$

where the global minimum is $f_3(\mathbf{x}^*) = 0$ and the minimizer is $\mathbf{x}^* = (0, \cdots, 0)^T$.

The Griewank function has many widespread and regularly distributed local minima. Thus, the test function is highly multimodal. Function values at local minima are very close to those at neighborhoods of the global minimum, making it one of the most difficult test functions.

The canonical and optimal PSO algorithms with the swarm size of $M = 5 \times 5 \times 5 = 125$ and the allowable swarm convergent tolerance of $\eta_a = 10^{-5}$ have been used to search within the three-dimensional space of $[-5, 5]^3$ for the global minimum of the Griewank function.

Simulations of both the canonical and the optimal PSO algorithms have been conducted at least 10 times each. Numerical test results are typically classified into two

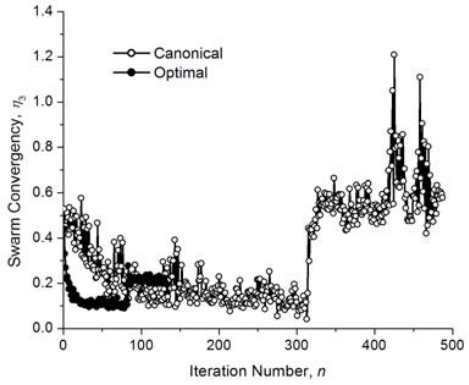

Fig. 5 Comparisons of swarm convergency between canonical and optimal PSO algorithms for the Griewank function

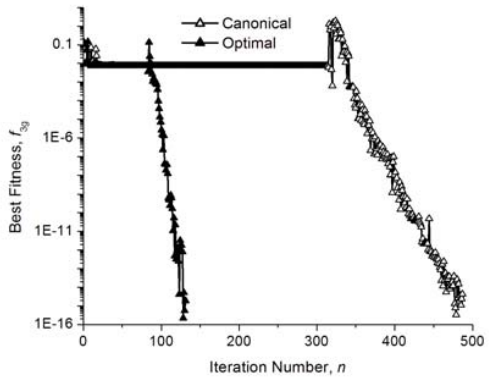

Fig. 6 Comparisons of best fitness between canonical and optimal PSO algorithms for the Griewank function

categories: not initially trapped in local minima and initially trapped in local minima. For not initially trapped-type of simulation results, only one result from each algorithm has been selected since most results are initially trapped in local minima. The optimal PSO algorithm yields the accurate results of $x_1 = x_2 = x_3 = 0.00000000$ within 43 iterations while the canonical algorithm takes 161 iterations to obtain the same precision of 8 decimal places.

For initially trapped-type of numerical results, the canonical PSO algorithm gives only two correct solutions with an average precision of 7 decimal places through an average total of 485 iterations, seven results remain trapped in local minima after 1000 iterations, and one result is totally trapped in local minima at 714 iterations. The optimal PSO algorithm produces all ten correct solutions with an average precision of 7 decimal places through an average 169 total iterations. For initially trapped-type of simulations, selected results for swarm convergency and best fitness are shown in Figs. 5 and 6, respectively. The canonical PSO algorithm escapes from local minima at iteration 315 and reaches the solution of $x_1 = 0.0000000$, $x_2 = 0.0000000$, and $x_3 = 0.0000000$ at iteration 487 while the optimal PSO algorithm escapes from local minima at iteration 83 and obtains the same result at iteration 131. Thus, the optimal PSO algorithm is more capable at solving optimization problems as difficult as the Griewank function than is the canonical PSO algorithm.

IV. Discussion

The Fibonacci sequence is the following series of numbers: $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \cdots$. Ratios of the Fibonacci sequence sequentially define the commonly used weightings. They are 0/1, 1/1, 1/2, 2/3, $\cdots$, and $(\sqrt{5}-1)/2$, which correspond to F_0/F_1 (the Euler forward weighting), F_1/F_2 (the Euler backward weighting), F_2/F_3 (the Crank-Nicolson weighting), F_3/F_4 (the Galerkin weighting), $\cdots$, and F_{k-1}/F_k as k approaches infinity (the optimal weighting). In the canonical PSO algorithm, the constants of $c_1 = c_2 = 2.05$ and $\chi = 0.72984$ are used. Thus, the canonical PSO algorithm used the Crank-Nicolson weightings between cognitive and social behaviors and between intensification searches and diversification searches. In the optimal PSO algorithm, the optimal weightings of $c_{o1} = (\sqrt{5}-1)/2$ and $c_{o2} = 1 - c_{o1}$ have been used for the cognitive factor and the social factor, respectively. By the same token, the optimal weightings of $\chi_o = (\sqrt{5}-1)/2$ and $1 - \chi_o$ have been applied for intensification searches and diversification searches, respectively.

The working principle of the optimal PSO algorithm can be demonstrated by the expected value of the position occupied by particle i at iteration n+1 updated from iteration n in two-dimensional scenarios shown in Fig. 7. At iteration n, the three positions $\mathbf{X}_i^n$, $\mathbf{P}_i^n$, and $\mathbf{P}_g^n$ and a velocity vector $\mathbf{V}_i^n$ are known. In the optimal PSO algorithm, the vector summation of 80.9% of the distance from $\mathbf{X}_i^n$ to $\mathbf{P}_i^n$ and 50.0% of the distance from $\mathbf{X}_i^n$ to $\mathbf{P}_g^n$ are expected to determine the intensification search vector and 38.2% of the magnitude from $\mathbf{V}_i^n$ constructs the diversification search vector. The expected intensification distance is calculated by its deterministic distance divided by 2 since the expectation of the uniform PDF is 0.5. The vector constructed from

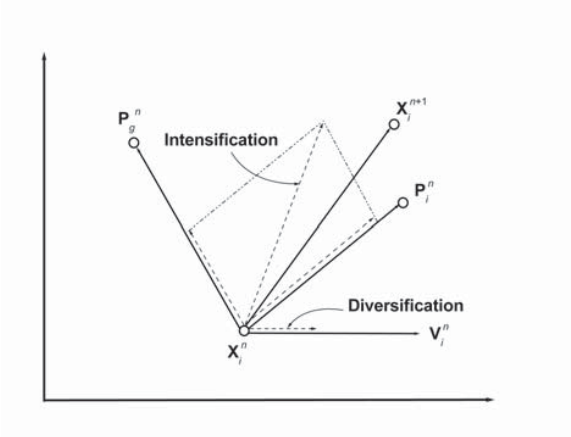

Fig. 7 Search mechanism for optimal PSO algorithm

$\mathbf{X}_i^n$ to $\mathbf{X}_i^{n+1}$ is determined by the vector summation of the intensification search vector and the diversification search vector shown in Fig. 7.

The optimal PSO algorithm is particularly accurate for cases in which particles of a swarm need to escape from local minima. Take, for example, the Griewank function. Particles for the two algorithms were initially trapped in local minima, the optimal particles escaped from local minima and found the global minimum location in all ten numerical simulations while the canonical PSO algorithm obtained the same accurate results in two out of ten numerical simulations. Because more weight is put on the cognitive behavior rather than the social behavior so that particles are moving toward individual best ever rather than the global best position in the optimal PSO algorithm. When trapped into local minima, on one hand, if more particles are moving toward the global best and the global best is also trapped at local minima, the particles tend to remain there. On the other hand, however, if more particles are moving toward their individual best positions, even though the global best is also trapped at local minima, more particles are still out of local minima to explore and the particles tend to help the global best escape from local minima. In general, because more weight is put on the intensification search over the diversification search so as to fine-tune solutions and to produce more accurate results in the optimal PSO algorithm.

The optimal PSO algorithm, compared to the canonical PSO algorithm, exhibits more stable swarm convergent behaviors in all three benchmark simulations shown in Figs. 1, 3, and 5 and more stable best fitness behaviors in all three benchmark simulations shown in Figs. 2, 4, and 6 since the canonical PSO algorithm generates more oscillation on both the swarm convergency and best fitness. The optimal PSO algorithm is more stable since the perturbation constant and the constriction constant are optimally balanced. Nevertheless, the factor for the diversification term in the optimal PSO algorithm is smaller than the constriction constant, which is used as a weight for both the diversification and the intensification terms in the canonical PSO algorithm.

For the de Jong function, the optimal PSO algorithm produces an average precision of 9 decimal places through an average of 57 total iterations while the canonical PSO algorithm needs an average of 202 total iterations. For the Rosenbrock function, the optimal PSO algorithm produces an average precision of 10 decimal places through an average of 228 total iterations while the canonical PSO algorithm needs an average of 478 total iterations. For the Griewank function, the optimal PSO algorithm produces an average precision of 7 decimal places through an average of 169 total iterations while the canonical PSO algorithm needs an average of 485 total iterations and they are calculated from two available results without considering the other 8 trapped results over 1000 iterations. Through all three bench mark functions, numerical results demonstrate that the optimal PSO algorithm is more efficient than the canonical PSO algorithm.

V. Conclusion

Ratios of the Fibonacci sequence systematically define commonly used weightings. The two asymptotic ratios of the Fibonacci sequence, F_{k-1}/F_k as k approaches infinity and F_{k-2}/F_k as k approaches infinity, are herein defined as the optimal weightings.

In the optimal PSO algorithm, the optimal weightings of $c_{o1} = (\sqrt{5}-1)/2$ and $c_{o2} = (3-\sqrt{5})/2$ have been used for the cognitive factor and the social factor, respectively. Similarly, $\chi_o = (\sqrt{5}-1)/2$ and $1-\chi_o$ have been applied for intensification searches and diversification searches, respectively. Nevertheless, the optimal perturbation constant of $\phi_o = 2+\sqrt{5}$ has been derived.

Through numerical simulations of the de Jong, the Rosenbrock, and the Griewank functions, the optimal PSO algorithm indeed achieves better numerical properties including accuracy, stability, and efficiency, and outperforms the canonical PSO algorithm.

References

[1] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," in Proc. IEEE Int. Conf. Neural Networks, Perth, Australia, Nov. 1995, pp. 1942–1948.
[2] R. C. Eberhart and J. Kennedy, "A New Optimizer Using Particle Swarm Theory," in Proc. 6th Int. Symp. Micromachine Human Science, Nagoya, Japan, Oct. 1995, pp. 39–43.
[3] J. Kennedy, R. C. Eberhart, and Y. Shi, Swarm Intelligence. San Francisco, CA: Morgan Kaufmann, 2001.
[4] Y. Shi and R. C. Eberhart, "A Modified Particle Swarm Optimizer," in Proc. IEEE Inter. Conf. Evol. Comput., Anchorage, AK, May 1998, pp. 69–73.
[5] M. Clerc and J. Kennedy, "The Particle Swarm–Explosion, Stability, and Convergence in a Multidimensional Complex Space," IEEE Trans. Evol. Comput., vol. 6, no. 1, pp. 58–73, Feb. 2002.
[6] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization–An overview," Swarm Intell., vol. 1, no. 1, pp. 33–57, Aug. 2007.
[7] K. E. Parsopoulos and M. N. Vrahatis, "Recent Approaches to Global Optimization Problems through Particle Swarm Optimization," Nat. Comput., vol. 1, no. 2-3, pp. 235–306, Jun. 2002.
[8] A. Banks, J. Vincent, and C. Anyakoha, "A Review of Particle Swarm Optimization. Part I: Background and Development," Nat. Comput., vol. 6, no. 4, pp. 467–484, Dec. 2007.
[9] A. Banks, J. Vincent, and C. Anyakoha, "A Review of Particle Swarm Optimization. Part II: Hybridisation, Combinatorial, Multicriteria and Constrained Optimization, and Indicative Applications," Nat. Comput., vol. 7, no. 1, pp. 109–124, Mar. 2008.
[10] Y. Valle, G. K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. G. Harley, "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems," IEEE Trans. Evol. Comput., vol. 12, no. 2, pp. 171–195, Apr. 2008.

Fuzhang Zhao received his Ph.D. degree in mechanical engineering and mechanics from Drexel University, Philadelphia, in 1998. He worked as a Postdoctoral Associate at the GRASP Lab at the University of Pennsylvania in 2001.

His research interests include applied mathematics, finite element methods, mechanics of materials, mechanical design, and analyses. He is currently very much interested in optimization methods, particularly in Particle Swarm Optimization algorithms.