Soft Comput
DOI 10.1007/s00500-016-2474-6

FOUNDATIONS

Particle swarm optimization algorithm: an overview

Dongshu Wang[1] · **Dapei Tan**[1] · **Lei Liu**[2]

Abstract Particle swarm optimization (PSO) is a population-based stochastic optimization algorithm motivated by intelligent collective behavior of some animals such as flocks of birds or schools of fish. Since presented in 1995, it has experienced a multitude of enhancements. As researchers have learned about the technique, they derived new versions aiming to different demands, developed new applications in a host of areas, published theoretical studies of the effects of the various parameters and proposed many variants of the algorithm. This paper introduces its origin and background and carries out the theory analysis of the PSO. Then, we analyze its present situation of research and application in algorithm structure, parameter selection, topology structure, discrete PSO algorithm and parallel PSO algorithm, multi-objective optimization PSO and its engineering applications. Finally, the existing problems are analyzed and future research directions are presented.

Communicated by A. Di Nola.

Electronic supplementary material The online version of this article (doi:10.1007/s00500-016-2474-6) contains supplementary material, which is available to authorized users.

✉ Dongshu Wang
wangdongshu@zzu.edu.cn

Dapei Tan
1581901458@qq.com

Lei Liu
luckyliulei@126.com

1 School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, Henan, China

2 Department of Research, The People's Bank of China, Zhengzhou Central Sub-Branch, Zhengzhou, China

Published online: 17 January 2017

Keywords Particle swarm optimization · Topology structure · Discrete PSO · Parallel PSO · Multi-objective optimization PSO

1 Introduction

Particle swarm optimization (PSO) algorithm is a stochastic optimization technique based on swarm, which was proposed by Eberhart and Kennedy (1995) and Kennedy and Eberhart (1995). PSO algorithm simulates animal's social behavior, including insects, herds, birds and fishes. These swarms conform a cooperative way to find food, and each member in the swarms keeps changing the search pattern according to the learning experiences of its own and other members.

Main design idea of the PSO algorithm is closely related to two researches: One is evolutionary algorithm, just like evolutionary algorithm; PSO also uses a swarm mode which makes it to simultaneously search large region in the solution space of the optimized objective function. The other is artificial life, namely it studies the artificial systems with life characteristics.

In studying the behavior of social animals with the artificial life theory, for how to construct the swarm artificial life systems with cooperative behavior by computer, Millonas proposed five basic principles (van den Bergh 2001):

(1) Proximity: the swarm should be able to carry out simple space and time computations.
(2) Quality: the swarm should be able to sense the quality change in the environment and response it.
(3) Diverse response: the swarm should not limit its way to get the resources in a narrow scope.
(4) Stability: the swarm should not change its behavior mode with every environmental change.

(5) Adaptability: the swarm should change its behavior mode when this change is worthy.

Note that the fourth principle and the fifth one are the opposite sides of the same coin. These five principles include the main characteristics of the artificial life systems, and they have become guiding principles to establish the swarm artificial life system. In PSO, particles can update their positions and velocities according to the environment change, namely it meets the requirements of proximity and quality. In addition, the swarm in PSO does not limit its movement but continuously search the optimal solution in the possible solution space. Particles in PSO can keep their stable movement in the search space, while change their movement mode to adapt the change in the environment. So particle swarm systems meet the above five principles.

2 Origin and background

In order to illustrate production background and development of the PSO algorithm, here we first introduce the early simple model, namely Boid (Bird-oid) model (Reynolds 1987). This model is designed to simulate the behavior of birds, and it is also a direct source of the PSO algorithm.

The simplest model can be depicted as follows. Each individual of the birds is represented by a point in the Cartesian coordinate system, randomly assigned with initial velocity and position. Then run the program in accordance with "the nearest proximity velocity match rule," so that one individual has the same speed as its nearest neighbor. With the iteration going on in the same way, all the points will have the same velocity quickly. As this model is too simple and far away from the real cases, a random variable is added to the speed item. That is to say, at each iteration, aside from meeting "the nearest proximity velocity match," each speed will be added with a random variable, which makes the total simulation to approach the real case.

Heppner designed a "cornfield model" to simulate the foraging behavior of a flock of birds (Clerc and Kennedy 2002). Assume that there was a "cornfield model" on the plane, i.e., food's location, and birds randomly dispersed on the plane at the beginning. In order to find the location of the food, they moved according to the following rules.

First, we assume that position coordinate of the cornfield is (x_0, y_0), and position coordinate and velocity coordinate of individual bird are (x, y) and (v_x, v_y), respectively. Distance between the current position and cornfield is used to measure the performance of the current position and speed. The further the distance to the "cornfield", the better the performance, on the contrary, the performance is worse. Assume that each bird has the memory ability and can memorize the best position it ever reached, denoted as $pbest$. a is velocity adjusting constant, $rand$ denotes a random number in [0,1], change in the velocity item can be set according to the following rules:

if $x > pbestx$, $v_x = v_x - rand \times a$, otherwise, $v_x = v_x + rand \times a$.

if $y > pbesty$, $v_y = v_y - rand \times a$, otherwise, $v_y = v_y + rand \times a$.

Then assume that the swarm can communicate in some way, and each individual is able to know and memorize the best location (marked as $gbest$) of the total swarm so far. And b is the velocity adjusting constant; then, after the velocity item was adjusted according to above rules, it must also update according to the following rules:

if $x > gbestx$, $v_x = v_x - rand \times b$, otherwise, $v_x = v_x + rand \times b$.

if $y > gbesty$, $v_y = v_y - rand \times b$, otherwise, $v_y = v_y + rand \times b$.

Computer simulation results show that when a/b is relatively large, all individuals will gather to the "cornfield" quickly; on the contrary, if a/b is small, the particles will gather around the "cornfield" unsteadily and slowly. Through this simple simulation, it can be found that the swarm can find the optimal point quickly. Inspired by this model, Kennedy and Eberhart devised an evolutionary optimization algorithm, after a sea of trials and errors, they finally fixed the basic algorithm as follows:

$$\begin{aligned} v_x &= v_x + 2 * rand * (pbestx - x) \\ &\quad + 2 * rand * (gbestx - x) \\ x &= x + v_x \end{aligned} \tag{1}$$

They abstracted each individual to be a particle without mass and volume, with only velocity and position, so they called this algorithm "particle swarm optimization algorithm."

On this basis, PSO algorithm can be summarized as follows: PSO algorithm is a kind of searching process based on swarm, in which each individual is called a particle defined as a potential solution of the optimized problem in D-dimensional search space, and it can memorize the optimal position of the swarm and that of its own, as well as the velocity. In each generation, the particles information is combined together to adjust the velocity of each dimension, which is used to compute the new position of the particle. Particles change their states constantly in the multi-dimensional search space, until they reach balance or optimal state, or beyond the calculating limits. Unique connection among different dimensions of the problem space is introduced via the objective functions. Many empirical evidences have showed that this algorithm is an effective optimization tool. Flowchart of the PSO algorithm is shown in Fig. 1.

The following gives a relatively complete presentation of the PSO algorithm. In the continuous space coordinate system, mathematically, the PSO can be described

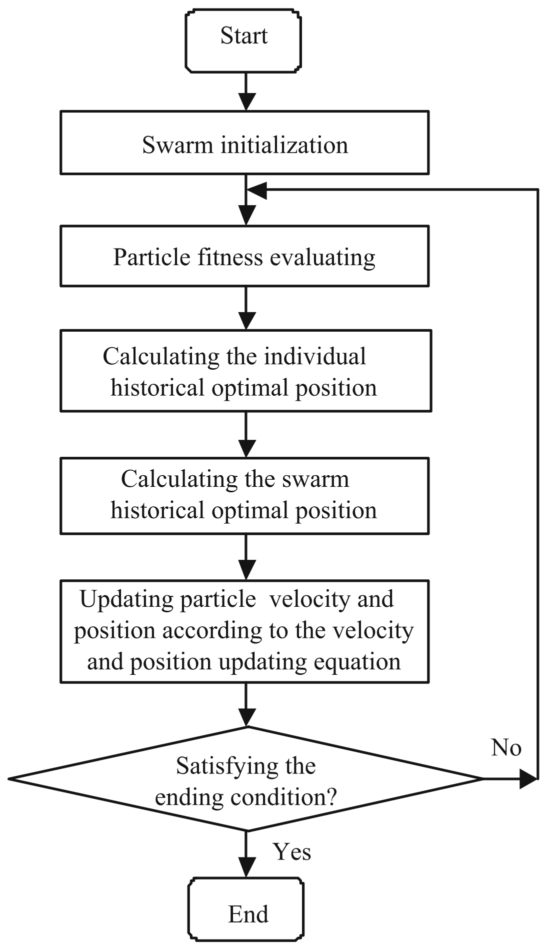

Fig. 1 Flowchart of the particle swarm optimization algorithm

as follows. Assume that swarm size is N, each particle's position vector in D-dimensional space is $X_i = (x_{i1}, x_{i2}, \cdots, x_{id}, \cdots, x_{iD})$, velocity vector is $V_i = (v_{i1}, v_{i2}, \cdots, v_{id}, \cdots, v_{iD})$, individual's optimal position (i.e., the optimal position that the particle has experienced) is $P_i = (p_{i1}, p_{i2}, \cdots, p_{id}, \cdots, p_{iD})$, swarm's optimal position (i.e., the optimal position that any individual in this swarm has experienced) is represented as $P_g = (p_{g1}, p_{g2}, \cdots, p_{gd}, \cdots, p_{gD})$. Without loss of generality, taking the minimizing problem as the example, in the initial version of the PSO algorithm, update formula of the individual's optimal position is:

$$p_{i,t+1}^{d} = \begin{cases} x_{i,t+1}^{d}, & \text{if } f(X_{i,t+1}) < f(P_{i,t}) \\ p_{i,t}^{d}, & \text{otherwise} \end{cases} \tag{2}$$

The swarm's optimal position is that of all the individual's optimal positions. Update formula of velocity and position is denoted as follows, respectively:

$$v_{i,t+1}^{d} = v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{g,t}^{d} - x_{i,t}^{d}) \tag{3}$$

$$x_{i,t+1}^{d} = x_{i,t}^{d} + v_{i,t+1}^{d} \tag{4}$$

Since the initial version of PSO was not very effective in optimization problem, a modified PSO algorithm (Shi and Eberhart 1998) appeared soon after the initial algorithm was proposed. Inertia weight was introduced to the velocity update formula, and the new velocity update formula became:

$$v_{i,t+1}^{d} = \omega * v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{g,t}^{d} - x_{i,t}^{d}) \tag{5}$$

Although this modified algorithm has almost the same complexity as the initial version, it has greatly improved the algorithm performance; therefore, it has achieved extensive applications. Generally, the modified algorithm is called canonical PSO algorithm, and the initial version is called original PSO algorithm.

By analyzing the convergence behavior of the PSO algorithm, Clerc and Kennedy (2002) introduced a variant of the PSO algorithm with constriction factor χ which ensured the convergence and improved the convergence rate. Then, the velocity update formula became:

$$v_{i,t+1}^{d} = \chi (v_{i,t}^{d} + \phi_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + \phi_2 * rand * (p_{g,t}^{d} - x_{i,t}^{d})) \tag{6}$$

Obviously, there is no essential difference between the iteration formulas (5) and (6). If appropriate parameters are selected, the two formulas are identical.

PSO algorithm has two versions, called global version and local version, respectively. In the global version, two extremes that the particles track are the optimal position *pbest* of its own and the optimal position *gbest* of the swarm. Accordingly, in local version, aside from tracking its own optimal position *pbest*, the particle does not track the swarm optimal position *gbest*, instead it tracks all particles' optimal position *nbest* in its topology neighborhood. For the local version, the velocity update equation (5) became:

$$v_{i,t+1}^{d} = \omega * v_{i,t}^{d} + c_1 * rand * (p_{i,t}^{d} - x_{i,t}^{d}) + c_2 * rand * (p_{l,t}^{d} - x_{i,t}^{d}) \tag{7}$$

where p_l was the optimal position in the local neighborhood.

In each generation, iteration procedure of any particle is illustrated in Fig. 2. Analyzing the velocity update formula from a sociological perspective, we can see that in this update formula, the first part is the influence of the particle's previous velocity. It means that the particle has confidence on its current moving state and conducts inertial moving according

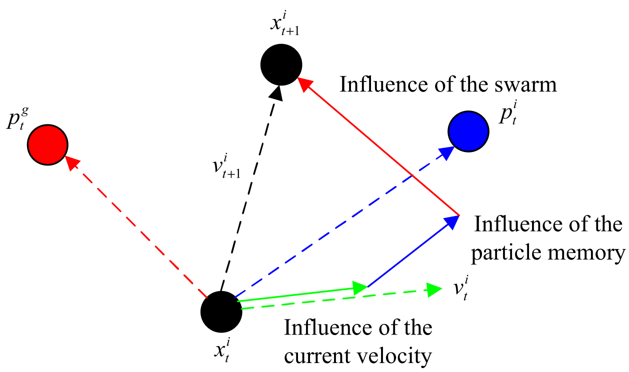

Fig. 2 Iteration scheme of the particles

to its own velocity, so parameter ω is called inertia weight. The second part depends on the distance between the particle's current position and its own optimal position, called the "cognitive" item. It means particle's own thinking, i.e., particle's move resulting from its own experience. Therefore, parameter c_1 is called cognitive learning factor (also called cognitive acceleration factor). The third part relies on the distance between the particle's current position and the global (or local) optimal position in the swarm, called "social" factor. It means the information share and cooperation among the particles, namely particle's moving coming from other particles' experience in the swarm. It simulates the move of good particle through the cognition, so the parameter c_2 is called social learning factor (also called social acceleration factor).

Due to its intuitive background, simple and easy to implement, as well as the wide adaptability to different kinds of functions, since the PSO algorithm has been proposed, it has obtained great attention. In the past twenty years, both the theory and application of the PSO algorithm have achieved great progress. Researchers have had a preliminary understanding of the principle, and its application has been realized in different domains.

PSO is a stochastic and parallel optimization algorithm. Its advantages can be summarized as follows: It does not require the optimized functions differential, derivative and continuous; its convergence rate is fast; and the algorithm is simple and easy to execute through programming. Unfortunately, it also has some disadvantages (Wang 2012): (1) For the functions with multiple local extremes, it probably falls into the local extreme and cannot get correct result. Two reasons result in this phenomenon: One is the characteristics of the optimized functions and the other is the particles' diversity disappearing quickly, causing premature convergence. These two factors are usually inextricably intertwined. (2) Due to lack of cooperation of good search methods, PSO algorithm cannot get satisfactory results. The reason is that the PSO algorithm does not sufficiently use the information obtained in the calculation procedure. During each iteration, instead it only uses the information of the swarm optima and individual optima. (3) Though PSO algorithm provides the possibility of global search, it cannot guarantee convergence to the global optima. (4) PSO algorithm is a meta-heuristic bionic optimization algorithm, and there is no rigorous theory foundation so far. It is designed only through simplifying and simulating the search phenomenon of some swarms, but it neither explains why this algorithm is effective from the principle, nor specifies its applicable range. Therefore, PSO algorithm is generally suitable for a class of optimization problems which are high dimensional and need not to get very accurate solutions.

Now there are many different kinds of researches about the PSO algorithm, and they can be divided into the following eight categories: (1) Theoretically analyze the PSO algorithm and try to understand its working mechanism. (2) Change its structure and try to get better performance. (3) Study the influence of various parameters configuration on PSO algorithm. (4) Study the influence of various topology structures on PSO algorithm. (5) Study the parallel PSO algorithm. (6) Study the discrete PSO algorithm. (7) Study the multi-objective optimization with the PSO algorithm. (8) Apply the PSO algorithm to various engineering fields. The remainder of this paper will begin to summarize the current researches on PSO algorithm from the above eight categories. Because the related studies are too much, we cannot do all well, so we just pick up some representative ones to review.

3 Theoretical analysis

Nowadays, theory study of the PSO algorithm mainly focuses on the principle of the PSO algorithm, i.e., how the particles to interact with each other, why it is effective for many optimization problems, but not obvious for other problems. Specifically, researches on this problem can be divided into three aspects: One is the moving trajectory of a single particle; another is the convergence problem; and the third is the evolution and distribution of the total particle system with the time.

The first analysis of simplified particles behavior was carried out by Kennedy (1998), who gave different particle trajectories under a series of design choices through simulating. The first theoretical analysis of simplified PSO algorithm was proposed by Ozcan and Mohan (1998) who indicated that in a simplified one-dimensional PSO system, a particle moved along a path defined by a sinusoidal wave, and determined its amplitude and frequency randomly. However, their analysis was merely limited to the simple PSO model without the inertia weight, and assumed that P_{id} and P_{gd} kept unchanged. Actually, P_{id} and P_{gd} changed frequently, so the particle' trajectory was a sine wave composed of many different amplitudes and frequencies. Therefore, the total trajectory looked still disorder. This reduced the effect of their conclusions significantly.

The first formal analysis of the PSO algorithm's stability was carried out by Clerc and Kennedy (2002), but this analysis method treated the random coefficients as constants, so it simplified the standard stochastic PSO to a deterministic dynamic system. The resulting system was a second-order linear dynamic system whose stability depended on the system poles or the eigenroots of state matrix. van den Bergh (2001) did a similar analysis of the deterministic version of PSO algorithm and determined the regions where the stability could be guaranteed in the parameter space. Convergence and parameters selection were also addressed in the literature (Trelea 2003; Yasuda et al. 2003). But the authors admitted that they did not take the stochastic feature of the PSO algorithm into account, thus their results had certain limitations. A similar analysis about the continuous version of PSO algorithm has also been done in Emara and Fattah (2004).

As it has already been proposed, the PSO algorithm adopts constant ω and uniform distribution random numbers c_1 and c_2. How the first- and second-order stability regions of the particle trajectories will be changed if ω also uses a random variable, and/or c_1 and c_2 conform to other statistical distributions instead of the uniform distribution? First-order stability analysis (Clerc and Kennedy 2002; Trelea 2003; Bergh and Engelbrecht 2006) aimed to test that stability of the mean trajectories relied on the parameters (ω, ϕ), where $\phi = (a_g + a_l)/2$, and c_1 and c_2 were uniform distribution in the intervals $[0, a_g]$ and $[0, a_l]$, respectively. Stochastic stability analysis contained higher-order moments and had been proved to be very useful for understanding the particle swarm dynamics and clarifying the PSO convergence properties (Fernandez-Martinez and Garcia-Gonzalo 2011; Poli 2009).

Bare Bones PSO (BBPSO) was proposed by Kennedy (2003) as a model of PSO dynamics. Its particle's velocity update conforms to a Gaussian distribution. Although Kennedy's original formulation is not competitive to standard PSO, adding a component-wise jumping mechanism, and a tuning of the standard deviation, can produce a comparable optimization algorithm. Hence, al Rifaie and Blackwell (2012) proposed a Bare Bones with jumps algorithm (BBJ), with an altered search spread component and a smaller jump probability. It used the difference between the neighborhood best with the current position rather than the difference between either the left and right neighbors' bests (in local neighborhood) or the particle's personal best and the neighborhood best (in global neighborhood). Three performance measures (i.e., accuracy, efficiency and reliability) were utilized to compare the BBJ with other standard PSO of Clerc–Kennedy and other variations of BBJ. Using these measures, it was shown that in terms of accuracy, when benchmarks with successful convergence were considered, the accuracy of BBJ compared to other algorithms was significantly better. Additionally, BBJ was empirically shown to be both the most efficient and the most reliable algorithm in both local and global neighborhoods.

Meanwhile, social variant of PSO ($a_l = 0$) and fully informed particle swarm (Mendes et al. 2004) were also studied by Poli (2008). Garcia-Gonza and Fernandez-Martinez (2014) presented the convergence and stochastic stability analysis of a series of PSO variants, and their research was different from the classical PSO in the statistical distribution of the three PSO parameters: inertia weight, local and global acceleration factors. They gave an analytical presentation for the upper limit of the second-order stability areas (the so called USL curves) of the particle trajectories, which is available for most of the PSO algorithms. Numerical experiments showed that the best algorithm performance could be obtained through tuning the PSO parameters close to the USL curve.

Kadirkamanathan et al. (2006) analyzed the stability of particle dynamics by using the Lyapunov stability analysis and the concept of passive system. This analysis did not assume that all parameters were non-random, and obtained the sufficient conditions of stability. It was based on the random particle dynamics that represented particle dynamics as a nonlinear feedback control system. Such system had a deterministic linear part and a nonlinear one and/or a time-varying gain in the feedback loop. Though it considered the influence of random components, its stability analysis was carried out aiming at the optimal position; therefore, the conclusion cannot be applied to non-optimal particles directly.

Even the original PSO algorithm could converge, it could only converge to the optima that the swarm could search, and could not guarantee that the achieved solution was the best, even it could not guarantee that it was the local optima. van den Bergh and Engelbrecht (2002) proposed a PSO algorithm which could ensure the algorithm convergence. It applied a new update equation for the global optimal particle and made it to generate a random search near the global optimal position, while other particles updated by their original equations. This algorithm could ensure the PSO algorithm to convergence to the local optimal solution with the cost of faster convergence rate, but its performance was inferior to the canonical PSO algorithm in multi-modal problems.

Lack of population diversity was regarded early (Kennedy and Eberhart 1995) as the important influence factor for premature convergence of the swarm toward a local optimum; hence, enhancing diversity was considered to be an useful approach to escape from the local optima (Kennedy and Eberhart 1995; Zhan et al. 2009). Enhancing the swarm diversity, however, is harmful to fast convergence toward the optimal solution. This phenomenon is well known because it was proved by Wolpert and Macready (1997) that an algorithm cannot surpass all the others on each kind of problem. Hence, research trials to promote the performance of an optimization algorithm should not be intended to search for a general function optimizer (Mendes et al. 2004; Wolpert and Macready 1997), but rather search for a general problem-solver which can perform well on many well-balanced practical benchmark problems (Garcia-Martinez and Rodriguez 2012).

Avoiding premature convergence on a local optimum solution, meanwhile keeping the fast convergence feature of the original PSO formulation, is an important reason why a few PSO variants have been put forward (Valle et al. 2008). These methods include fine-tuning the PSO parameters to manipulate the particle velocity updating (Nickabadi et al. 2011), various PSO local formulation to consider the best solution within a local topological particle neighborhood instead of the entire swarm (Kennedy and Mendes 2002, 2003; Mendes et al. 2004) and integrating the PSO with other heuristic algorithms (Chen et al. 2013). For example, comprehensive learning PSO (Liang et al. 2006) applied a new learning scheme to increase the swarm diversity in order to avoid the premature convergence in solving multi-modal problems. ALC-PSO (Chen et al. 2013) endowed the swarm leader an increasing age and a lifespan to escape from the local optima and thus avoid the premature convergence. Self-regulating PSO (Tanweer et al. 2016) adopted a self-regulating inertia weight and self-perception on the global search direction to get faster convergence and better results.

For the spherically symmetric local neighborhood functions, Blackwell (2005) has theoretically analyzed and experimentally verified the speed features with diversity loss in PSO algorithm. Kennedy (2005) has systematically studied how the speed influence the PSO algorithm, and it was helpful to understand the contribution of the speed to the PSO performance. Clerc (2006) studied the iteration process of the PSO at the stagnant stage, as well as the roles of each random coefficient in detail; finally, he gave the probability density functions of each random coefficient.

4 Algorithm structure

There are a sea of enhancement approaches for the PSO algorithm structure, which can be classified into 8 main subsections as follows.

4.1 Adopting multi-sub-populations

In 2001, Suganthan (1999) introduced the concept of subpopulation of the genetic algorithm and brought a reproduction operator in the PSO algorithm. Dynamic multi-swarm PSO was proposed by Liang and Suganthan (2005) where the swarm was divided into several sub-swarm, and these sub-swarms were regrouped frequently to share information among them. Peng and Chen (2015) presented a symbiotic particle swarm optimization (SPSO) algorithm to optimize the neural fuzzy networks. The presented SPSO algorithm used the multi-swarm strategy which used each particle to represent a single fuzzy rule and each particle in each swarm evolved separately to avoid falling into a local optima. Chang (2015) proposed a modified PSO algorithm to solve multi-modal function optimization problems. It divided the original single swarm into several sub-swarms based on the order of particles. The best particle in each sub-swarm was recorded and then applied into the velocity updating formula to replace the original global best particle in the whole population. To update all particles in each sub-swarm, the enhanced velocity formula was adopted.

In addition, Tanweer et al. (2016) presented a new dynamic mentoring and self-regulation-based particle swarm optimization (DMeSR-PSO) algorithm which divided the particles into mentor, mentee and independent learner groups according to their fitness differences and Euclidian distances with respect to the best particle. Performance of DMeSR-PSO had been extensively evaluated on 12 benchmark functions (unimodal and multi-modal) from CEC2005, more complex shifted and rotated CEC2013 benchmark functions and 8 real-world optimization problems from CEC2011. The performance of DMeSR-PSO on CEC2005 benchmark functions had been compared with six PSO variants and five meta-heuristic algorithms. The results clearly highlighted that DMeSR-PSO provided the most consistent performance on the selected benchmark problems. The non-parametric Friedman test followed by the pair-wise post hoc Bonferroni–Dunn test provided the evidence that DMeSR-PSO was statistically better than the selected algorithms with 95% confidence level. Further, the performance had also been statistically compared with seven PSO variants on the CEC2013 benchmark functions where the performance of DMeSR-PSO was significantly better than five algorithms with a confidence level of 95%. The performance of DMeSR-PSO on the CEC2011 real-world optimization problems was better than the winner and runner-up of the competition, indicating that DMeSR-PSO was an effective optimization algorithm for real-world applications. Based on these results, the DMeSR-PSO would be recommended to deal with the CEC test sets.

For the high-dimensional optimization problem, PSO algorithm requires too many particles which results in high computational complexity; thus, it is difficult to achieve a satisfactory solution. So recently, cooperative particle swarm algorithm (CPSO-H) (Bergh and Engelbrecht 2004) was proposed which split the input vector into multiple sub-vector, and for each sub-vector, a particle swarm was used to optimize it. Although the CPSO-H algorithm used one-dimensional swarm to search for each dimension, respectively, after the search results were integrated by a global swarm, its performance on multi-modal problems had been greatly improved. Further, Niu et al. (2005) introduced master–slave sub-population mode into the PSO algorithm and put forward a multi-population cooperative PSO algorithm. Similarly, Seo et al. (2006) proposed a multi-grouped PSO which used N groups of particles to search N peaks of the multi-modal problems simultaneously. Selleri et al. (2006) used multiple independent sub-populations and added some new items to the particle velocity update formula which made the particles to move toward the historical optimal position of the sub-population, or away from the gravity center of other sub-populations.

4.2 Improving the selection strategy for particle learning object

Al-kazemi and Mohan (2002) proposed a multi-phase PSO algorithm in which particles were grouped according to the temporary search targets in different phases, and these temporary targets allowed the particles to move toward or away from its own or the global best position. Ting et al. (2003) modified every particle's $pBest$, and every dimension learned from randomly determined other dimensions. After that, if the new $pBest$ was better, then it was used to replace the original $pBest$.

In PSO algorithm, Yang and Wang (2006) introduced the roulette selection technique to determine the $gBest$, so that in the early stage of evolution, all individuals had chance to lead the search direction to avoid premature. Zhan et al. (2011) introduced an orthogonal learning PSO in which an orthogonal learning scheme was used to get efficient exemplars. Abdelbar et al. (2005) proposed a fuzzy measure, and several particles with the best fitness values in each neighbor could affect other particles.

Contrary to the original PSO algorithm, there is a class method which makes the particles to move away from the worst position instead of toward the best position. Yang and Simon (2005) proposed to record the worst position rather than the best position in the algorithm, and all particles moved away from these worst positions. Similarly, Leontitsis et al. (2006) introduced a new concept—repel operator which used the information of individual's optimal position and swarm's optimal position. Meanwhile, it also recorded the current individual's worst positions and swarm's worst positions and used them to repel the particles toward the best position, so that the swarm could reach the best position quickly.

4.3 Modifying particle's update formula

Many of these methods use chaotic sequences to modify the particle positions, in which particles search for solutions extensively due to the chaoticity. It is known that these PSO variants have a more diverse search than the standard PSO (Tatsumi et al. 2013). Coelho and Lee (2008) randomized the cognitive and social behaviors of the swarm with chaotic sequences and Gaussian distribution, respectively. Tatsumi et al. (2015) emphasized the chaotic PSO to exploit a virtual quartic objective function according to the personal and global optima. This model adopted a perturbation-based chaotic system derived from a quartic tentative objective function through applying the steepest descent method with a perturbation. The function was determined for each particle, and it had two global minima at the $pbest$ of the particle and the $gbest$.

In addition to these methods, in the Bare bones PSO algorithm (Kennedy 2003), particle positions were updated by using a Gaussian distribution. Since many foragers and wandering animals followed a Levy distribution of steps, this distribution was useful for optimization algorithms. So Richer and Blackwell (2006) replaced the particle dynamics within PSO with random sampling from a Levy distribution. A range of benchmark problems were utilized to test its performance; the resulting Levy PSO performed as well, or better, than a standard PSO or equivalent Gaussian models. Moreover, in speed update equation, Hendtlass (2003) added memory ability to each particle and He et al. (2004) introduced passive congregation mechanism. Zeng et al. (2005) introduced acceleration term into the PSO algorithm which changed the PSO algorithm from a second-order stochastic system into a third-order stochastic one. In order to improve the global search ability of the PSO algorithm, Ho et al. (2006) proposed a new speed and position update formula and introduced the variable "age." Moreover, Ngoa et al. (2016) proposed an improved PSO to enhance the performances of standard PSO by using a new movement strategy for each particle. Particles in the PSO fly to their own predefined target instead of the best particles (i.e., personal and global bests). Performance of proposed improved PSO was illustrated by applying it to 15 unconstrained (i.e., unimodal and multi-modal) benchmarks and 15 computationally expensive unconstrained benchmarks.

4.4 Modifying velocity update strategy

Although PSO performance has improved over the past decades, how to select suitable velocity updating strategy and parameters remains an important research domain. Ardizzon et al. (2015) proposed a novel example of the original particle swarm concept, with two types of agents in the swarm, the "explorers" and the "settlers", that could dynamically exchange their role during the search procedure. This approach can dynamically update the particle velocities at each time step according to the current distance of each particle from the best position found so far by the swarm. With good exploration capabilities, uniform distribution random numbers in the velocity updating strategy may also affect the particle moving. Thus, Fan and Yan (2014) put forward a self-adaptive PSO with multiple velocity strategies (SAPSO-MVS) to enhance PSO performance. SAPSO-MVS could generate self-adaptive control parameters in the total evolution procedure and adopted a novel velocity updating scheme to improve the balance between the exploration and exploitation capabilities of the PSO algorithm and avoided to tune the PSO parameters manually. Roy and Ghoshal (2008) proposed Crazy PSO in which particle velocity was randomized within predefined limits. Its aim was to randomize the velocity of some particles, named as "crazy particles" through using a predefined probability of craziness to keep the diversity for global search and better convergence. Unfortunately, values of the predefined probability of craziness could only be obtained after a few experiments. Peram et al. (2003) presented a fitness–distance ratio-based PSO (FDR-PSO), in which a new velocity updating equation was used to regenerate the velocity of each particle. Li et al. (2012) presented a self-learning PSO in which a velocity update scheme could be automatically modified in the evolution procedure. Lu et al. (2015b) proposed a mode-dependent velocity updating equation with Markovian switching parameters in switching PSO to overcome the contradiction between the local search and the global search, which made it easy to jump out of the local minimum.

Liu et al. (2004) argued that too frequent velocity update would weaken the particles' local exploit ability and decrease the convergence, so he proposed a relaxation velocity update strategy, which updated the speed only when the original speed could not improve the particle's fitness value further. Experimental results proved that this strategy could reduce the computation load greatly and accelerate the convergence. Diosan and Oltean (2006) used genetic algorithm to evolve PSO algorithm structure, i.e., particles updating order and frequency.

4.5 Modifying the speed or position constrain method and dynamically determining the search space

Chaturvedi et al. (2008) dynamically controlled the acceleration coefficients in maximum and minimum limits. Determining the bound value of the acceleration coefficients, however, was a very difficult issue because it needed several simulations. Stacey et al. (2003) offered a new speed constrain method to re-randomize the particle speed and a novel position constrain method to re-randomize the particle position. Clerc (2004) brought a contraction-expansion coefficient into evolution algorithms to ensure the algorithm convergence, while relaxing the speed bound. Other approaches, such as squeezing the search space (Barisal 2013), had also been proposed to dynamically determine the search space.

4.6 Combining PSO with other search techniques

It has two main purposes: One is to increase the diversity and avoid premature; the other is to improve the local search ability of the PSO algorithm. In order to promote the search diversity in the PSO, a sea of models have been studied (Poli et al. 2007). These hybrid algorithms included introducing various genetic operators to the PSO algorithm, such as selection (Angeline 1998a, b; Lovbjerg et al. 2001), crossover (Angeline 1998b; Chen et al. 2014), mutation (Tsafarakis et al. 2013) or Cauchy mutation (Wang et al. 2011) to increase the diversity and improve its ability to escape from the local minima. Meng et al. (2015) proposed a new hybrid optimization algorithm called crisscross search particle swarm optimization (CSPSO), which was different from PSO and its variants in that its every particle was directly expressed by *pbest*. Its population was updated by modified PSO and crisscross search optimization in sequence during each iteration. Seventeen benchmark functions (including four unimodal functions, five multi-modal functions and several complicated shifted and rotated functions) were used to test the feasibility and efficiency of the CSPSO algorithm, but it had not coped with the premature convergence in the later period of the optimization. Vlachogiannis and Lee (2009) presented a novel control equation in enhanced coordinated aggregation PSO for better communication among particles to improve the local search. It permitted the particles to interact with its own best experience as well as all other particles with better experience on aggregate basis, instead of the global optimal experience. Selvakumar and Thanushkodi (2009) presented civilized swarm optimization (CSO), through combining society-civilization algorithm with PSO to enhance its communication. This new algorithm provided clustered search which produced better exploration and exploitation of the search space. Unfortunately, it needed several experiments to decide the optimal control parameters of the CSO.

Lim and Isa (2015) put forward a hybrid PSO algorithm which introduced the fuzzy reasoning and a weighted particle to construct a new search behavior model to increase the search ability of the conventional PSO algorithm. Besides the information of the global best and individual best particles, Shin and Kita (2014) took advantage of the information of the second global best and second individual best particles to promote the search performance of the original PSO.

Tanweer et al. (2016) presented a novel particle swarm optimization algorithm named as self-regulating particle swarm optimization (SRPSO) algorithm which introduced the best human learning schemes to search the optimum results. The SRPSO used two learning schemes. The first scheme adopted a self-regulating inertia weight, and the second one adopted the self-perception on the global search direction. Other methods or models to improve the diversity included: attracting-exclusion model (Riget and Vesterstrom 2002), predator-prey model (Gosciniak 2015), uncorrelative component analysis model (Fan et al. 2009), dissipative model (Xie et al. 2002), self-organizing model (Xie et al. 2004), life cycle model (Krink and Lovbjerg 2002), Bayesian optimization model (Monson and Seppi 2005), chemical reaction optimization (Li et al. 2015b), neighborhood search mechanism (Wang et al. 2013), collision-avoiding mechanism (Blackwell and Bentley 2002), information sharing mechanism (Li et al. 2015a), local search technique (Sharifi et al. 2015), cooperative behavior (Bergh and Engelbrecht 2004), hierarchical fair competition (Chen et al. 2006b), external memory (Acan and Gunay 2005), gradient descent technique (Noel and Jannett 2004), simplex method operator (Qian et al. 2012; El-Wakeel 2014), hill climbing method (Lin et al. 2006b), division of labor (Lim and Isa 2015), principal component analysis (Mu et al. 2015), Kalman filtering (Monson and Seppi 2004), genetic algorithm (Soleimani and Kannan 2015), shuffled frog leaping algorithm (Samuel and Rajan 2015), random search algorithm (Ciuprina et al. 2007), Gaussian local search (Jia et al. 2011), simulated annealing (Liu et al. 2014; Geng et al. 2014), taboo search (Wen and Liu 2005), Levenberg–Marquardt algorithm (Shirkhani et al. 2014), ant colony algorithm (Shelokar et al. 2007), artificial bee colony (Vitorino et al. 2015; Li et al. 2011), chaos algorithm (Yuan et al. 2015), differential evolution (Zhai and Jiang 2015), evolutionary programming (Jamian et al. 2015), multi-objective cultural algorithm (Zhang et al. 2013). PSO algorithm was also extended in quantum space by Sun et al. (2004). The novel PSO model was based on the delta potential well and modeled the particles as having quantum behaviors. Furthermore, Medasani and Owechko (2005) expanded the PSO algorithm through introducing the possibility of c-means and probability theory, and put forward probabilistic PSO algorithm.

4.7 Improving for multi-modal problems

The seventh solution is specifically for multi-modal problems, hoping to find several better solutions. In order to obtain several better solutions for the optimized problem, Parsopoulos and Vrahatis (2004) used deflection, stretching and repulsion and other techniques to find as many as possible minimum points by preventing the particles from moving to the minimum area ever found before. However, this method would generate new local optima at both ends of the detected local ones, which might lead the optimization algorithm to fall into local optima. Therefore, Jin et al. (2005) proposed a new form of function transformation which could avoid this disadvantage.

Another variant is a niche PSO algorithm proposed by Brits et al. (2003), to locate and track multiple optima by using multiple sub-populations simultaneously. Brits et al. (2002) also studied a method to find the multiple optimal solutions simultaneously through adjusting the fitness value calculating way. On the basis of the niche PSO algorithm, Schoeman and Engelbrecht (2005) adopted vector operation to determine the candidate solution and its border in each niche through using vector dot production operation and paralleled this process to obtain better results. However, there was a common problem in each niche PSO algorithm, namely it needed to determine a niche radius, and the algorithm performance was very sensitive to the niche radius. In order to solve this problem, Benameur et al. (2006) presented an adaptive method to determine the niching parameters.

4.8 Keeping diversity of the population

Population diversity is especially important for enhancing the global convergence of the PSO algorithm. The easiest way to keep population diversity was resetting some particles or total particle swarm when the diversity was very small. Lovbjerg and Krink (2002) adopted a self-organized criticality in PSO algorithm to depict the proximity degree among the particles in the swarm, further, to decide whether re-initialize the particle positions or not. Clerc (1999) presented a deterministic algorithm named as Re-Hope, when the search space was quite small but had not yet found solutions (No-Hope); then, it reset the swarm. To keep the population diversity and balance the global and local search, Fang et al. (2016) proposed a decentralized form of quantum-inspired particle swarm optimization (QPSO) with cellular structured population (called cQPSO). Performance of cQPSO-*lbest* was investigated on 42 benchmark functions with different properties (including unimodal, multi-modal, separated, shifted, rotated, noisy, and mis-scaled) and compared with a set of PSO variants with different topologies and swarm-based evolutionary algorithms (EAs).

The modified PSO of Park et al. (2010) introduced chaotic inertia weight which decreased and oscillated simultaneously under the decreasing line in a chaotic manner. In this manner, additional diversity was brought into the PSO but it needed tuning the chaotic control parameters. Recently, Netjinda et al. (2015) presented a novel mechanism into PSO to increase the swarm diversity, a mechanism inspired by the collective response behavior of starlings. This mechanism is composed of three major steps: initialization, which prepared alternative populations for the next steps; identifying seven nearest neighbors; and orientation changed which updated the particle velocity and position according to those neighbors and chosen the best alternative. Due to this collective response mechanism, the Starling PSO realized a wider scope of the search space and hence avoided suboptimal solutions. The trade-off for the improving performance was that this algorithm added more processes to the original algorithm. As a result, more parameters were needed while the additional process, the collective response process, also made this algorithm consume more execution time. However, the algorithm complexity of the Starling PSO was still the same as that of the original PSO.

5 Parameters selection

There are several important parameters in PSO algorithm, i.e., inertia weight ω (or constriction factor χ), learning factors c_1 and c_2, speed limits V_{max}, position limits X_{max}, swarm size and the initial swarm. Some researchers fixed other parameters and only studied the influence of single parameter on the algorithm, while some researchers also studied the effect of multiple parameters on the algorithm.

5.1 Inertia weight

Current studies believe that the inertia weight has the greatest influence on the performance of PSO algorithm, so there are the most researches in this area. Shi and Eberhart (1998) was the first individual to discuss the parameter selection in PSO. They brought an inertia efficient ω into the PSO and promoted the convergence feature. An extension of this work adopted fuzzy systems to nonlinearly change the inertia weight during optimization (Shi and Eberhart 2001).

Generally, it is believed that in PSO, inertia weight is used to balance the global search and the local search, and bigger inertia weight is tended to global search while the smaller inertia weight is tended to local search, so the value of inertia weight should gradually reduce with the time. Shi and Eberhart (1998) suggested that inertia weight should be set to [0.9, 1.2] and a linearly time-decreasing inertia weight could significantly enhance the PSO performance.

As the fixed inertia weight usually cannot get satisfactory results, there appeared some PSO variants whose inertia weight declined linearly along with iteration times (Shi and Eberhart 1998), adaptive changed (Nickabadi et al. 2011), adjusted by a quadratic function (Tang et al. 2011) and by the population information (Zhan et al. 2009), adjusted based on Bayesian techniques (Zhang et al. 2015), exponential decreasing inertia weight strategy (Lu et al. 2015a), declined according to the nonlinear function (Chatterjee and Siarry 2006), and Sugeno function (Lei et al. 2005) in search process. At the same time, there are many methods that the inertia weight adaptively changes along with some evaluation indexes, such as the successful history of search (Fourie and Groenwold 2002), evolution state (Yang et al. 2007), particle average velocity (Yasuda and Iwasaki 2004), population diversity (Jie et al. 2006), smoothness change in the objective function (Wang et al. 2005), evolutionary speed and aggregation degree of the particle swarm, individual search ability (Qin et al. 2006). Even, Liu et al. (2005) determined whether accept the inertia weight change or not according to Metropolis criteria.

Some people also adopted a random inertia weight, such as setting to [0.5+(rnd/2.0)] (Eberhart and Shi 2001), [0, 1] uniform distribution random numbers (Zhang et al. 2003). Jiang and Bompard (2005) introduced the chaos mechanism in selecting the inertia weight, so the inertia weight could traverse [0, 1]. The modified PSO of Park et al. (2010) introduced chaotic inertia weight which oscillated and decreased simultaneously under the decreasing line in a chaotic way, but it needed tuning the chaotic control parameters.

5.2 Learning factors c_1 and c_2

The learning factors c_1 and c_2 represent the weights of the stochastic acceleration terms that pull each particle toward $pBest$ and $gBest$ (or $nBest$). In many cases, c_1 and c_2 are set to 2.0 which make the search to cover the region centered in $pBest$ and $gBest$. Another common value is 1.49445 which can ensure the convergence of PSO algorithm (Clerc and Kennedy 2002). After a lot of experiments, Carlisle and Dozier (2001) put forward a better parameters set which set c_1 and c_2 to 2.8 and 1.3, respectively, and the performance of this setting was further confirmed by Schutte and Groenwold (2005). Inspired by the idea of time-varying inertia weight, there appeared many PSO variants whose learning factors changed with time (Ivatloo 2013), such as learning factor linearly decreased with time (Ratnaweera et al. 2004), dynamically adjusted based on the particles' evolutionary states (Ide and Yasuda 2005), dynamically adjusted in accordance with the number of the fitness values deteriorate persistently and the swarm's dispersion degree (Chen et al. 2006a).

In most cases, the two learning factors c_1 and c_2 have the same value, so that the social and cognitive search has the same weight. Kennedy (1997) studied two kinds of extremes: model with only the social term and with only the cognitive term, and the result showed that these two parts were very crucial to the success of swarm search, while there were no definitive conclusions about the asymmetric learning factor.

There were researches which determined the inertia weight and learning factors simultaneously. Many researchers adopted various optimization techniques to dynamically determine the inertia weight and learning factors, such as genetic algorithm (Yu et al. 2005), adaptive fuzzy algorithm (Juang et al. 2011), differential evolutionary algorithm (Parsopoulos and Vrahatis 2002b), adaptive critic design technology (Doctor and Venayagamoorthy 2005).

5.3 Speed limits V_{max}

Speed of the particles was constrained by a maximum speed V_{max} which can be used as a constraint to control the global search ability of the particle swarm. In original PSO algorithm, $\omega = 1, c_1 = c_2 = 2$, particles' speed often quickly increases to a very high value which will affect the performance of the PSO algorithm, so it is necessary to restrict particle velocity. Later, Clerc and Kennedy (2002) pointed out that it was not necessary to restrict the particle velocity, introducing constriction factor to the speed update formula could also realize the purpose of limiting particle velocity. However, even the constriction factor was used, experiments showed that better result would be obtained if the particle velocity was limited simultaneously (Eberhart and Shi 2000), so the idea of speed limitation was still retained in PSO algorithm. Generally speaking, V_{max} was set to the dynamic range of each variable, and usually a fixed value, but it could also linearly decrease with time (Fan 2002) or dynamically reduce according to the success of search history (Fourie and Groenwold 2002).

5.4 Position limits X_{max}

Positions of the particles can be constrained by a maximum position X_{max} that can avoid particles flying out of the physical solution space. Robinson and Rahmat-Samii (2004) put forward three different control techniques, namely absorbing wall, reflecting wall and invisible wall. Once one of the dimensions of a particle hit the boundary of the solution space, the absorbing wall set the velocity in that corresponding dimension to zero, while the reflecting wall changed the direction of particle velocity, the particle was eventually pulled back to the allowable solution space by the two walls. In order to reduce the calculation time and avoid affect the motions of other particles, the invisible walls did not calculate the fitness values of the particles flying out of the boundary. However, the performance of PSO algorithm was greatly influenced by the dimension of the problem and the relative position between the global optima and the search space boundary. In order to solve this problem, Huang and Mohan (2005) integrated the characteristics of absorbing wall and reflecting wall and proposed a hybrid damping boundary, to obtain robust and consistent performance. And Mikki and Kishk (2005) combined the techniques of hard position limit, absorbing wall and reflecting wall, and the result showed that it could obtain better results.

5.5 Population size

Selection of population size is related to the problems to be solved, but it is not very sensitive to the problems. Common selection is 20–50. In some cases, larger population is used to meet the special needs.

5.6 Initialization of the population

Initialization of the population is also a very important problem. Generally, the initial population is randomly generated, but there are also many intelligent population initialization methods, such as using the nonlinear simplex method (Parsopoulos and Vrahatis 2002a), centroidal Voronoi tessellations (Richards and Ventura 2004), orthogonal design (Zhan et al. 2011), to determine the initial population of PSO algorithm, making the distribution of the initial population as evenly as possible, and help the algorithm to explore the search space more effectively and find a better solution. Robinson et al. (2002) pointed out that the PSO algorithm and GA algorithm could be used in turn, i.e., taking the population optimized by the PSO algorithm as the initial population of the GA algorithm, or conversely, taking the population optimized by GA algorithm as the initial population of the PSO algorithm, both methods could get better results. Yang et al. (2015) presented a new PSO approach called LHNPSO, with low-discrepancy sequence initialized particles and high-order ($1/\pi^2$) nonlinear time-varying inertia weight and constant acceleration coefficients. Initial population was produced through applying the Halton sequence to fill the search space adequately.

Furthermore, parameters of PSO algorithm could be adjusted by the methods such as sensitivity analysis (Bartz-Beielstein et al. 2002), regression trees (Bartz-Beielstein et al. 2004a) and calculate statistics (Bartz-Beielstein et al. 2004b), to promote the performance of PSO algorithm for solving the practical problems.

Besides these, Beheshti and Shamsuddin (2015) presented a nonparametric particle swarm optimization (NP-PSO) to improve the global exploration and the local exploitation in PSO without tuning algorithm parameters. This method integrated local and global topologies with two quadratic interpolation operations to enhance the algorithm search capacity.

6 Topological structure

Many researchers have proposed different population topology structures in the PSO algorithm because performance of PSO is directly influenced by population diversity. Therefore, designing different topologies to improve the performance of PSO algorithm is also an active research direction.

Since the topology is studied, it must be related to the concept of neighborhood. Neighborhood can be static, or may be dynamically determined. There are two ways to determine the neighborhood: One is determined according to the flag of the particles (or index) which has nothing to do with distance; the other is determined in accordance with the topological distance between the particles.

Most researches (Bratton and Kennedy 2007; Kennedy and Mendes 2002; Li 2010) used the static topology structure. Kennedy (1999) and Kennedy and Mendes (2002, 2003) had analyzed different kinds of static neighborhood structures and their influences on the performance of PSO algorithm, and regarded that adaptability of the star, ring and von Neumann topology were best. As studied by the work of Kennedy, PSO with a small neighborhood might have a better performance on complicated problems, while PSO with a large neighborhood would perform better on simple problems. Based on K-means clustering algorithm, Kennedy (2000) also proposed another version of local PSO algorithm, called social convergence method, with the hybrid space neighborhood and ring topology. Each particle updated itself by using the common experience of spatial clustering that it belonged to, rather than the experience of its own. Kennedy (2004) proved enhanced performance of the PSO algorithm through applying ring topology and made the particles to move in accordance with the normally distributed random perturbations. Engelbrecht et al. (2005) studied the ability of the basic PSO algorithm to locate and maintain several solutions in the multi-modal optimization problems and found that the global neighborhood PSO ($gBest$ PSO) was incapable of this problem, while the efficiency of the local neighborhood PSO ($nBest$ PSO) was very low.

Mendes et al. (2004) presented the fully informed particle swarm algorithm that used the information of entire neighborhood to guide the particles to find the best solution. Influence of each particle on its neighbor was weighted by its fitness value and the neighborhood size. Peram et al. (2003) developed a new PSO algorithm based on fitness–distance ratio (FDR-PSO) using the interaction of the neighbors. In updating each dimension component of the velocity, FDR-PSO algorithm selected a $nBest$ of other particles with higher fitness and much closer to the particle to be updated. This algorithm selected different neighborhood particles in updating each dimension of the speed, so it was more effective than that only selected one neighbor particle in all speed dimensions. Peer et al. (2003) used different neighborhood topologies to study the performance of guaranteed convergence PSO (GCPSO).

There is also a small part of the researches about dynamic topology. Lim and Isa (2014) presented a novel PSO variant named as PSO with adaptive time-varying topology connectivity (PSO-ATVTC) which used an ATVTC module and a new learning scheme. The presented ATVTC module particularly aimed to balance the algorithm's exploration/exploitation search through changing the particle's topology connection with time in accordance with its search performance. Suganthan (1999) used a dynamically adjusted neighborhood, and this neighborhood could increase gradually until it included all the particles. Hu and Eberhart (2002) studied a dynamic neighborhood, and in each generation, the nearest m particles were selected to be the new neighbors of one particle. Lin et al. (2006a) studied two kinds of dynamically randomized neighborhood topology. Mohais et al. (2005) presented a PSO algorithm with area of influence (AOI), in AOI, influence of the optimal particle on other particles depended on the distances between them. Hierarchical PSO (Hanaf et al. 2016) used the dynamic tree hierarchy based on the performance of each particle in population to define the neighborhood structure.

All above neighborhood topologies were used to determine the group experience $gBest$, while Hendtlass (2003) used neighborhood topology to determine individual experience $pBest$.

7 Discrete PSO

A sea of optimization problems involve discrete or binary variables, and the typical examples include scheduling problems or routing problems. While the update formula and procedure of the PSO algorithm are originally designed for the continuous space, which limited its application in discrete optimization domains, therefore it need some changes to adapt to the discrete space.

In continuous PSO, trajectories are defined as changes in position on a number of dimensions. By contrast, binary particle swarm optimization (BPSO) trajectories are changes in the probability that a coordinate will take on a value of zero or one.

Jian et al. (2004) defined the first discrete binary version of PSO to optimize the structure of neural networks. The particles used binary strings to encode. By using the sigmoid function, the velocity was limited to [0,1], and it was interpreted as "the change in probability." By re-defining the particle position and velocity, continuous PSO could be changed to discrete PSO to solve discrete optimization problems. Tang et al. (2011) extended this method into quantum space. Ratnaweera et al. (2004) and Afshinmanesh et al. (2005) presented the discrete PSO further.

In addition, a few modified binary PSO algorithms have been proposed. An angle modulation PSO (AMPSO) was proposed by Mohais et al. (2005) to produce a bit string to solve the original high-dimensional problem. In this approach, the high-dimensional binary problem is reduced to a four-dimensional problem defined in continuous space, with a direct mapping back to the binary space by angle modulation. al Rifaie and Blackwell (2012) presented a new discrete particle swarm optimization method for the discrete time-cost trade-off problem. Two large-scale benchmark problems were use to evaluate the performance of the DPSO. The results indicated that DPSO provided an effective and robust alternative for solving real-world time-cost optimization problems. However, the large-scale benchmark problems used in this study included up to 630 activities and up to five time-cost alternatives and might have certain limitations for representing the complexity of some large-scale construction projects. Peer et al. (2003) developed a genetic BPSO model without fixing the size of the swarm. In this algorithm, two operations, i.e., birth and death, were introduced to dynamically modulate the swarm. Because the birth and death rates changed naturally with time, this new model permitted oscillations in the size of the swarm. So it was a more natural simulation of the social behaviors of the intelligent animals. An enhanced binary BPSO was presented by Kadirkamanathan et al. (2006) that introduced the concepts of genotype-phenotype representation and the mutation operator of GA into the BPSO. A novel BPSO was proposed to overcome the BPSO problems by Lu et al. (2015b). Although the performance of the algorithm was better than that of the BPSO, the new BPSO may be trapped into the local optimum and generated premature convergence. Therefore, two different methods were designed to prevent the stagnation of the new BPSO. One of the methods improved the performance of the new BPSO by introducing the concept of guaranteed convergence BPSO. Another method, modified new BPSO, adopted the mutation operator to avoid the stagnation issue. Chatterjee and Siarry (2006) developed an essential binary PSO to optimize problems in the binary search spaces. In this algorithm, PSO was divided into its essential elements, and alternative explanations of these elements were proposed. Previous direction and state of each particle ware also considered to search for the good solutions for the optimized problems. Fourie and Groenwold (2002) presented a fuzzy discrete particle swarm optimization to cope with real-time charging coordination of plug-in electric vehicles. Wang et al. (2005) presented a binary bare bones PSO to search for optimal feature selection. In this algorithm, a reinforced memory scheme was developed to modify the local leaders of particles to prevent the degradation of distinguished genes in the particles, and an uniform combination was presented to balance the local exploitation and the global exploration of the algorithm.

Traditional PSO suffers from the dimensionality problem, i.e., its performance deteriorates quickly when the dimensionality of the search space increases exponentially, which greatly limits its application to large-scale global optimization problems. Therefore, for large-scale social network clustering, Brits et al. (2002) presented a discrete PSO algorithm to optimize community structures in social networks. Particle position and velocity were redefined in terms of a discrete form. Subsequently, the particle modify strategies were redesigned in accordance with the network topology.

Discrete PSO (DPSO) had been successfully applied to many discrete optimization tasks, such as Sudoku puzzle (Liu et al. 2004), multi-dimensional knapsack problems (Banka and Dara 2015), jobshop scheduling problems (Vitorino et al. 2015), complex network clustering problems (Brits et al. 2002), optimizing the echo state network (Shin and Kita 2014), image matching problems (Qian et al. 2012), instance selection for time series classification (van den Bergh and Engelbrecht 2002), ear detection (Emara and Fattah 2004), feature selection (Chen et al. 2006b), capacitated location routing problem (Liang et al. 2006), generation maintenance scheduling problem (Schaffer 1985), elderly day care center timetabling (Lee et al. 2008), high-dimensional feature selection, classification and validation (Ardizzon et al. 2015), and high-order graph matching (Fang et al. 2016). All these problems had the irrespective challenges and were difficult to be optimized, but they could be effectively solved by DPSO.

Most of the above discrete PSO algorithms were indirect optimization strategies which determined the binary variables based on the probability rather than the algorithm itself; therefore, it could not make full use of the performance of PSO algorithm. In dealing with the integer variable, PSO algorithm was very easy to fall into local minimum. Original PSO algorithm learned from the experience of individual and its companions, the discrete PSO algorithm should also follow this idea. Based on the traditional velocity-displacement updating operation, Engelbrecht et al. (2005) analyzed the optimization mechanism of PSO algorithm and proposed the general particle swarm optimization (GPSO) model which was suitable to solve the discrete and combinational optimization problems. Nature of the GPSO model still conformed to the PSO mechanism, but its particle updating strategy could be designed either according to the features of the optimized problems, or integrating with other methods. Based on the similar ideas, Fan et al. (2009) defined local search and path-relinking procedures as the velocity operator to solve the traveling salesman problem. Beheshti and Shamsuddin (2015) presented a memetic binary particle swarm optimization strategy in accordance with the hybrid local and global searches in BPSO. This binary hybrid topology particle swarm optimization algorithm had been used to solve the optimization problems in the binary search spaces.

8 Multi-objective optimization PSO

In recent years, multi-objective (MO) optimization has become an active research area. In the multi-object optimization problems, each target function can be optimized independently and then find the optimal value for each target. Unfortunately, due to the conflicting among the objects, it is almost impossible to find a perfect solution for all the objectives. Therefore, only the Pareto optimal solution can be found.

Information sharing mechanism in PSO algorithm is very different from other optimization tools based on swarm. In the genetic algorithm (GA), the chromosomes exchange information with each other through crossover operation, which is a bidirectional information sharing mechanism. While in most PSO algorithms, only $gBest$ (or $nBest$) provides information for other particles. Due to the point attracting feature, traditional PSO algorithm cannot simultaneously locate multiple optimal points constituting the Pareto frontier. By giving different weights to all the objective functions, then combining them and running many times, though we can obtain multiple optimal solutions, we still want to find a method which can simultaneously obtain a group of Pareto optimal solutions.

In the PSO algorithm, a particle is an independent agent which can search the problem space according to the experience of its own and its companions. As mentioned above, the former is the cognitive part of the particles update formula, and the latter is the social part. Both parts play key role in guiding the particles' search. Therefore, choosing the appropriate social and cognitive guide ($gBest$ and $pBest$) is the key problems of the MOPSO algorithm. Selection of the cognitive guide conforms the same rule as that of the traditional PSO algorithm, and the only difference is that the guide should be determined in accordance with Pareto dominance. Selection of the social guide includes two steps: The first step is creating a candidate pool used to select the guide. In traditional PSO algorithm, the guide is selected from the $pBest$ of the neighbors. While in the MOPSO algorithm, the usual method is using an external pool to store more Pareto optimal solutions. The second step is choosing the guide. Selection of the $gBest$ should satisfy the following two standards: First, it should be able to provide effective guidance for the particles to obtain a better convergence speed; second, it needs to provide balanced search along the Pareto frontier, to maintain the diversity of the population. Two methods are usually used to determine the social guide: (1) Roulette selection mode which selects randomly in accordance with some standards, to maintain the diversity of the population. (2) Quantity standard: determine the social guide according to some procedures without involving random selection.

After Peram et al. (2003) presented the vector-evaluated genetic algorithm, an ocean of multi-objective optimization algorithms were proposed one after another, such as NSGA-II (Coello et al. 2004). Liu et al. (2016) were the first to study the application of the PSO algorithm in multi-objective optimization and emphasized the importance of the individual search and swarm search, but they did not adopt any method to maintain the diversity. On the basis of the concept of non-dominated optimal, Clerc (2004) used an external archive to store and determine which particles would be the non-dominated members, and these members were used to guide other particle's flight. Kennedy (2003) adopted the main mechanism of the NSGA-II algorithm to determine local optimal particle among the local optimal particles and their offspring particles, and proposed non-dominated sorting PSO which used the max-min strategy in the fitness function to determine the Pareto dominance. Moreover, Goldbarg et al. (2006) also used the non-dominated sorting PSO to optimize a U-tube steam generator mathematical model in the nuclear power plant.

Ghodratnama et al. (2015) applied the comprehensive learning PSO algorithm combining with Pareto dominance to solve the multi-objective optimization problems. Ozcan and Mohan (1998) developed an elitist multi-objective PSO that combined the elitist mutation coefficient to improve the particles' exploitation and exploration capacity. Wang et al. (2011) proposed an iterative multi-objective particle swarm optimization-based control vector parameterization to cope with the dynamic optimization of the state constrained chemical and biochemical engineering problems. In recent researches, Clerc and Kennedy (2002), Fan and Yan (2014), Chen et al. (2014), Lei et al. (2005), et al. also proposed the corresponding multi-objective PSO algorithms.

Among them, Li (2004) proposed a novel Cultural MOQPSO algorithm, in which cultural evolution mechanism was introduced into quantum-behaved particle swarm optimization to deal with multi-objective problems. In Cultural MOQPSO, the exemplar positions of each particle were obtained according to "belief space," which contained different types of knowledge. Moreover, to increase population diversity and obtain continuous and even-distributed Pareto fronts, a combination-based update operator was proposed to update the external population. Two quantitative measures, inverted generational distance and binary quality metric, were adopted to evaluate its performance. A comprehensive comparison of Cultural MOQPSO with some state-of-the-art evolutionary algorithms on several benchmark test functions, including ZDT, DTLZ and CEC2009 test instances, demonstrated that Cultural MOQPSO had better performance than the other MOQPSOs and MOPSOs. Besides, Cultural MOQPSO was also compared to 11 state-of-the-art evolutionary algorithms, by testing on the first 10 functions defined in CEC-2009. The comparative results demonstrated that, for half of the test functions, Cultural MOQPSO performed better than most of the 11 algorithms. According to these quantitate comparisons, the Cultural MOQPSO can be recommended to cope with the multi-objective optimization problems.

Because the fitness calculation consumes much computational resource, in order to reduce the calculation cost, it needs to reduce the evaluating numbers of fitness function. Pampara et al. (2005) adopted fitness inheritance technique and estimation technique to achieve this goal and compared the effect of fifteen kinds of inheritance techniques and four estimation techniques that were applied to multi-objective PSO algorithm.

There are two main methods to maintain the diversity of the MOPSO: Sigma method (Lovbjerg and Krink 2002) and ε-dominance method (Juang et al. 2011; Robinson and Rahmat-Samii 2004). Robinson and Rahmat-Samii (2004) put forward a multi-swarm PSO algorithm which broke down the whole swarm into three equally sized sub-swarms. Each sub-swarm applied different mutation coefficients, and this scheme enhanced the search capacity of the particles.

Due to the page limit, engineering applications of the PSO are attached in the supplementary file, interested readers are encouraged to refer it.

9 Noise and dynamic environments

State of dynamic system changes frequently, even continuously. Many practical systems involve the dynamic environment. For example, due to changes caused by the priority of customers, unexpected equipment maintenance, most of calculating time in the scheduling system was used to reschedule. In real applications, these changes in system states often needed to be re-optimized.

Using the PSO algorithm to track the dynamic system was initially proposed by Brits et al. (2003), and it followed the dynamic system through periodically resetting all particles' memories. Deb and Pratap (2002) also adopted the similar idea. After that, Geng et al. (2014) introduced an adaptive PSO algorithm, which could automatically track the changes in the dynamic system, and different environment detection and response techniques were tested on the parabolic benchmark function. It effectively increased the tracking ability for the environment change through testing the best particle in the swarm and reinitializing the particles. Later, Carlisle and Dozier (2000) used a random point in the search space to determine whether the environment changed or not, but it required centralized control, and it was inconsistent with PSO algorithm's distributed processing model. So Clerc (2006) proposed a Tracking Dynamical PSO (TDPSO) that made the fitness value of the best history position to decrease with time; thus, it did not need the centralized control. In order to respond to the rapidly changing dynamic environment, Binkley and Hagiwara (2005) added a penalty term in particles' update formula to keep the particles lying in an expanding swarm, and this method need not to examine whether the best point changed or not.

Experiments of Monson and Seppi (2004) showed that the basic PSO algorithm could work in the noisy environment efficiently and stably; even in many cases, the noise could also help PSO algorithm to avoid falling to the local optima. Moreover, Mostaghim and Teich (2003) also experimentally studied the performance of unified particle swarm algorithm in the dynamic environment. Nickabadi et al. (2011) proposed a anti-noise PSO algorithm. Pan proposed an effective hybrid PSO algorithm named PSOOHT by introducing the optimal computing budget allocation (OCBA) technique and hypothesis test, to solve the function optimization in the noisy environment.

Research objects of the above works are the simple dynamic systems, experiment functions used are the simple single-mode functions, and the changes are uniform ones in simple environment (that is, fixed step). In fact, the real dynamic systems are often nonlinear, and they change nonuniformly in complex multi-mode search space. Kennedy (2005) used four PSO (a standard PSO, two randomized PSO algorithms, and a fine-grained PSO) models to comparatively study a series of different dynamic environments.

10 Numerical experiments

PSO was also used in different numerical experiments. To handle the imprecise operating costs, demands and the capacity data in a hierarchical production planning system, Carlisle and Dozier (2001) used a modified variant of a possibilistic environment-based particle swarm optimization approach to solve an aggregate production plan model which used the strategy of simultaneously minimizing the most possible value of the imprecise total costs, maximizing the possibility of obtaining lower total costs and minimizing the risk of obtaining higher total costs. This method provides a novel approach to consider the natural uncertainty of the parameters in an aggregate production plan problem and can be applied in ambiguous and indeterminate circumstances of real-world production planning and scheduling problems with ill-defined data. To analyze the effects of process parameters (cutting speed, depth of cut and feed rate) on machining criteria, Ganesh et al. (2014) applied the PSO to optimize the cutting conditions for the developed response surface models. PSO program gave the minimum values of the considered criteria and the corresponding optimal cutting conditions. Lu used an improved PSO algorithm which adopted a combined fitness function to solve the squared error between the measured values and the modeled ones in system identification problems. Numerical simulations with five benchmark functions were used to validate the feasibility of PSO, and furthermore, numerical experiments were also carried out to evaluate the performance of the improved PSO. Consistent results demonstrated that combined fitness function-based PSO algorithm was feasible and efficient for system identification and could achieve better performance over conventional PSO algorithm.

To test the Starling PSO, eight numerical benchmarking functions which represent various characteristics of typical problems as well as a real-world application involving data clustering were used by Lu et al. (2015a). Experimental results showed that Starling PSO improved the performance of the original PSO and yielded the optimal solution in many numerical benchmarking functions and most of the real-world problems in the clustering experiments.

Selvakumar and Thanushkodi (2009) put forward an improved CPSO-VQO with a modified chaotic system whose bifurcation structure was irrelevant to the difference vector, and they also proposed a new stochastic method that selected the updating system according to the ratio between the components of the difference vector for each particle, and restarting and acceleration techniques to develop the standard updating system used in the proposed PSO model. Through numerical experiments, they verified that the proposed PSOs, PSO-TPC, PSO-SPC and PSO-SDPC were superior to the relatively simple existing PSOs and CPSO-VQO in finding high-quality solutions for various benchmark problems. Since the chaotic system used in these PSOs was based on the gbest and pbest, this search was mainly restricted around the two points in spite of its chaoticity.

In addition, Sierra and Coello (2005) conducted numerical experiments with benchmark objective functions with high dimensions to verify the convergence and effectiveness of the proposed initialization of PSO. Salehian and Subraminiam (2015) adopted an improved PSO to optimize the performance in terms of the number of alive nodes in wireless sensor networks. The performance of the adopted improved PSO was validated by the numerical experiments in conventional background.

11 Conclusions and discussion

As a technique appearing not long time, PSO algorithm has received wide attentions in recent years. Advantages of PSO algorithm can be summarized as follows: (1) It has excellent robustness and can be used in different application environments with a little modification. (2) It has strong distributed ability, because the algorithm is essentially the swarm evolutionary algorithm, so it is easy to realize parallel computation. (3) It can converge to the optimization value quickly. (4) It is easy to hybridize with other algorithms to improve its performance.

After many years development, the optimization speed, quality and robustness of the PSO algorithm have been greatly improved. However, the current studies mostly focus on the algorithm's implementation, enhancement and applications, while relevant fundamental research is far behind the algorithm's development. Lacking of mathematical theory basis greatly limits the further generalization, improvement and application of the PSO algorithm.

PSO algorithm research still exist a lot of unsolved problems, including but not limited to:

(1) Random convergence analysis. Although the PSO algorithm has been proved to be effective in real applications and has achieved some preliminary theoretical results, it has not provided the mathematical proofs about the algorithm convergence and convergence rate estimation so far.
(2) How to determine the algorithm parameters. Parameters in PSO are usually determined depending on the specific problems, application experience and numerous experiment tests, so it has no versatility. Hence, how to determine the algorithm parameters conveniently and effectively is another urgent problem to be studied.
(3) Discrete/binary PSO algorithm. Most research literatures reported in this paper deal with continuous variables. Limited research illustrated that the PSO algorithm had some difficulties in dealing with the discrete variables.
(4) Aiming at the characteristics of different problems, designing corresponding effective algorithm is a very meaningful work. For the specific application problems, we should deeply study PSO algorithm and extend its application from the breadth and depth. At the same time, we should focus on the highly efficient PSO design, combining the PSO with optimized problem or rules, PSO with the neural network, fuzzy logic, evolutionary algorithm, simulated annealing, taboo search, biological intelligence, and chaos, etc, to cope with the problem that the PSO is easy to be trapped into the local optima.
(5) PSO algorithm design research. More attention should be emphasized on the highly efficient PSO algorithm and put forward suitable core update formula and effective strategy to balance the global exploration and local exploitation.
(6) PSO application search. Nowadays, most of the PSO applications are limited in continuous, single objective, unconstrained, deterministic optimization problems, so we should emphasize the applications on discrete, multi-objective, constrained, un-deterministic, dynamic optimization problems. At the same time, the application areas of PSO should be expanded further.

Acknowledgements The authors thank the reviewers for their valuable comments/suggestions which helped to improve the quality of this paper significantly.

Compliance with ethical standards

Funding This study was funded by National Natural Sciences Funds of China (Grant Number 61174085).

Conflict of interest All the authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

References

Abdelbar AM, Abdelshahid S, Wunsch DCI (2005) Fuzzy pso: a generalization of particle swarm optimization. In: Proceedings of 2005 IEEE international joint conference on neural networks (IJCNN '05) Montreal, Canada, July 31–August 4, pp 1086–1091

Acan A, Gunay A (2005) Enhanced particle swarm optimization through external memory support. In: Proceedings of 2005 IEEE congress on evolutionary computation, Edinburgh, UK, Sept 2–4, pp 1875–1882

Afshinmanesh F, Marandi A, Rahimi-Kian A (2005) A novel binary particle swarm optimization method using artificial immune system. In: Proceedings of the international conference on computer as a tool (EUROCON 2005) Belgrade, Serbia, Nov 21–24, pp 217–220

Al-kazemi B, Mohan CK (2002) Multi-phase generalization of the particle swarm optimization algorithm. In: Proceedings of 2002 IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, August 7–9, pp 489–494

al Rifaie MM, Blackwell T (2012) Bare bones particle swarms with jumps ants. Lect Notes Comput Sci Ser 7461(1):49–60

Angeline PJ (1998a) Evolutionary optimization versus particle swarm optimization philosophy and performance difference. In: Evolutionary programming, Lecture notes in computer science, vol. vii edition. Springer, Berlin

Angeline PJ (1998b) Using selection to improve particle swarm optimization. In: Proceedings of the 1998 IEEE international conference on evolutionary computation, Anchorage, Alaska, USA, May 4–9, pp 84–89

Ardizzon G, Cavazzini G, Pavesi G (2015) Adaptive acceleration coefficients for a new search diversification strategy in particle swarm optimization algorithms. Inf Sci 299:337–378

Banka H, Dara S (2015) A hamming distance based binary particle swarm optimization (HDBPSO) algorithm for high dimensional feature selection, classification and validation. Pattern Recognit Lett 52:94–100

Barisal AK (2013) Dynamic search space squeezing strategy based intelligent algorithm solutions to economic dispatch with multiple fuels. Electr Power Energy Syst 45:50–59

Bartz-Beielstein T, Parsopoulos KE, Vrahatis MN (2002) Tuning pso parameters through sensitivity analysis. Technical Report CI 124/02, SFB 531. University of Dortmund, Dortmund, Germany, Department of Computer Science

Bartz-Beielstein T, Parsopoulos KE, Vegt MD, Vrahatis MN (2004a) Designing particle swarm optimization with regression trees. Technical Report CI 173/04, SFB 531. University of Dortmund, Dortmund, Germany, Department of Computer Science

Bartz-Beielstein T, Parsopoulos KE, Vrahatis MN (2004b) Analysis of particle swarm optimization using computational statistics. In: Proceedings of the international conference of numerical analysis and applied mathematics (ICNAAM 2004), Chalkis, Greece, pp 34–37

Beheshti Z, Shamsuddin SM (2015) Non-parametric particle swarm optimization for global optimization. Appl Soft Comput 28:345–359

Benameur L, Alami J, Imrani A (2006) Adaptively choosing niching parameters in a PSO. In: Proceedings of genetic and evolutionary computation conference (GECCO 2006), Seattle, Washington, USA, July 8–12, pp 3–9

Binkley KJ, Hagiwara M (2005) Particle swarm optimization with area of influence: increasing the effectiveness of the swarm. In: Proceedings of 2005 IEEE swarm intelligence symposium (SIS 2005), Pasadena, California, USA, June 8–10, pp 45–52

Blackwell TM (2005) Particle swarms and population diversity. Soft Comput 9(11):793–802

Blackwell TM, Bentley PJ (2002) Don't push me! Collision-avoiding swarms. In: Proceedings of IEEE congress on evolutionary computation, Honolulu, HI, USA, August 7–9, pp 1691–1697

Bratton D, Kennedy J (2007) Defining a standard for particle swarm optimization. In: Proceedings of the 2007 IEEE swarm intelligence symposium (SIS2007), Honolulu, HI, USA, April 19–23, pp 120–127

Brits R, Engelbrecht AP, van den Bergh F (2002) Solving systems of unconstrained equations using particle swarm optimization. In: Proceedings of IEEE international conference on systems, man, and cybernetics, hammamet, Tunisia, October 6–9, 2002. July 27–28, 2013, East Lansing, Michigan, pp 1–9

Brits R, Engelbrecht AP, van den Bergh F (2003) Scalability of niche PSO. In: Proceedings of the IEEE swarm intelligence symposium, Indianapolis, Indiana, USA, April 24–26, pp 228–234

Carlisle A, Dozier G (2000) Adapting particle swarm optimization to dynamic environments. In: Proceedings of the international conference on artificial intelligence, Athens, GA, USA, July 31–August 5, pp 429–434

Carlisle A, Dozier G (2001) An off-the-shelf PSO. In: Proceedings of the workshop on particle swarm optimization, Indianapolis, Indiana, USA

Chang WD (2015) A modified particle swarm optimization with multiple subpopulations for multimodal function optimization problems. Appl Soft Comput 33:170–182

Chatterjee A, Siarry P (2006) Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization. Comput Oper Res 33:859–871

Chaturvedi KT, Pandit M, Shrivastava L (2008) Self-organizing hierarchical particle swarm optimization for non-convex economic dispatch. IEEE Trans Power Syst 23(3):1079–1087

Chen J, Pan F, Cai T (2006a) Acceleration factor harmonious particle swarm optimizer. Int J Autom Comput 3(1):41–46

Chen K, Li T, Cao T (2006b) Tribe-PSO: a novel global optimization algorithm and its application in molecular docking. Chemom Intell Lab Syst 82:248–259

Chen W, Zhang J, Lin Y, Chen N, Zhan Z, Chung H, Li Y, Shi Y (2013) Particle swarm optimization with an aging leader and challenger. IEEE Trans Evolut Comput 17(2):241–258

Chen Y, Feng Y, Li X (2014) A parallel system for adaptive optics based on parallel mutation PSO algorithm. Optik 125:329–332

Ciuprina G, Ioan D, Munteanu I (2007) Use of intelligent-particle swarm optimization in electromagnetics. IEEE Trans Manag 38(2):1037–1040

Clerc M (1999) The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation (CEC 1999), pp 1951–1957, Washington, DC, USA, July 6–9, 1999

Clerc M (2004) Discrete particle swarm optimization. In: Onwubolu GC (ed) New optimization techniques in engineering. Springer, Berlin

Clerc M (2006) Stagnation analysis in particle swarm optimisation or what happens when nothing happens. Technical Report CSM-460, Department of Computer Science, University of Essex, Essex, UK, August 5–8, 2006

Clerc M, Kennedy J (2002) The particle swarm-explosion, stability and convergence in a multi dimensional complex space. IEEE Trans Evolut Comput 6(2):58–73

Coelho LDS, Lee CS (2008) Solving economic load dispatch problems in power systems using chaotic and gaussian particle swarm optimization approaches. Electr Power Energy Syst 30:297–307

Coello CAC, Pulido G, Lechuga M (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evolut Comput 8(3):256–279

Deb K, Pratap A (2002) A fast and elitist multi objective genetic algorithm: NSGA-II. IEEE Trans Evolut Comput 6(2):182–197 del Valle Y, Venayagamoorthy GK, Mohagheghi S, Hernandez JC, Harley RG (2008) Particle swarm optimization: basic concepts, variants and applications in power systems. IEEE Trans Evolut Comput 12:171–195

Diosan L, Oltean M (2006) Evolving the structure of the particle swarm optimization algorithms. In: Proceedings of European conference on evolutionary computation in combinatorial optimization (EvoCOP2006), pp 25–36, Budapest, Hungary, April 10–12, 2006

Doctor S, Venayagamoorthy GK (2005) Improving the performance of particle swarm optimization using adaptive critics designs. In: Proceedings of 2005 IEEE swarm intelligence symposium (SIS 2005), pp 393–396, Pasadena, California, USA, June 8–10, 2005

Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the 6th international symposium on micro machine and human science, pp 39–43, Nagoya, Japan, Mar 13–16, 1995

Eberhart RC, Shi Y (2000) Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the IEEE congress on evolutionary computation (CEC 2000), pp 84–88, San Diego, CA, USA, July 16–19, 2000

Eberhart RC, Shi Y (2001) Particle swarm optimization: developments, applications and resources. In: Proceedings of the IEEE congress on evolutionary computation (CEC 2001), pp 81–86, Seoul, Korea, May 27–30

El-Wakeel AS (2014) Design optimization of pm couplings using hybrid particle swarm optimization-simplex method (PSO-SM) algorithm. Electr Power Syst Res 116:29–35

Emara HM, Fattah HAA (2004) Continuous swarm optimization technique with stability analysis. In: Proceedings of American Control Conference, pp 2811–2817, Boston, MA, USA, June 30–July 2, 2004

Engelbrecht AP, Masiye BS, Pampard G (2005) Niching ability of basic particle swarm optimization algorithms. In: Proceedings of 2005 IEEE Swarm Intelligence Symposium (SIS 2005), pp 397–400, Pasadena, CA, USA, June 8–10, 2005

Fan H (2002) A modification to particle swarm optimization algorithm. Eng Comput 19(8):970–989

Fan Q, Yan X (2014) Self-adaptive particle swarm optimization with multiple velocity strategies and its application for p-xylene oxidation reaction process optimization. Chemom Intell Lab Syst 139:15–25

Fan SKS, Lin Y, Fan C, Wang Y (2009) Process identification using a new component analysis model and particle swarm optimization. Chemom Intell Lab Syst 99:19–29

Fang W, Sun J, Chen H, Wu X (2016) A decentralized quantum-inspired particle swarm optimization algorithm with cellular structured population. Inf Sci 330:19–48

Fernandez-Martinez JL, Garcia-Gonzalo E (2011) Stochastic stability analysis of the linear continuous and discrete PSO models. IEEE Trans Evolut Comput 15(3):405–423

Fourie PC, Groenwold AA (2002) The particle swarm optimization algorithm in size and shape optimization. Struct Multidiscip Optim 23(4):259–267

Ganesh MR, Krishna R, Manikantan K, Ramachandran S (2014) Entropy based binary particle swarm optimization and classification for ear detection. Eng Appl Artif Intell 27:115–128

Garcia-Gonza E, Fernandez-Martinez JL (2014) Convergence and stochastic stability analysis of particle swarm optimization variants with generic parameter distributions. Appl Math Comput 249:286–302

Garcia-Martinez C, Rodriguez FJ (2012) Arbitrary function optimisation with metaheuristics: no free lunch and real-world problems. Soft Comput 16:2115–2133

Geng J, Li M, Dong Z, Liao Y (2014) Port throughput forecasting by MARS-RSVR with chaotic simulated annealing particle swarm optimization algorithm. Neurocomputing 147:239–250

Ghodratnama A, Jolai F, Tavakkoli-Moghaddamb R (2015) Solving a new multi-objective multiroute flexible flow line problem by multi-objective particle swarm optimization and nsga-ii. J Manuf Syst 36:189–202

Goldbarg EFG, de Souza GR, Goldbarg MC (2006) Particle swarm for the traveling salesman problem. In: Proceedings of European conference on evolutionary computation in combinatorial optimization (EvoCOP2006), pp 99-110, Budapest, Hungary, April 10–12, 2006

Gosciniak I (2015) A new approach to particle swarm optimization algorithm. Expert Syst Appl 42:844–854

Hanaf I, Cabrerab FM, Dimanea F, Manzanaresb JT (2016) Application of particle swarm optimization for optimizing the process parameters in turning of peek cf30 composites. Procedia Technol 22:195–202

He S, Wu Q, Wen J (2004) A particle swarm optimizer with passive congregation. BioSystems 78:135–147

Hendtlass T (2003) Preserving diversity in particle swarm optimisation. In: Proceedings of the 16th international conference on industrial engineering applications of artificial intelligence and expert systems, pp 31–40, Loughborough, UK, June 23–26, 2003

Ho S, Yang S, Ni G (2006) A particle swarm optimization method with enhanced global search ability for design optimizations of electromagnetic devices. IEEE Trans Magn 42(4):1107–1110

Hu X, Eberhart RC (2002) Adaptive particle swarm optimization: Detection and response to dynamic systems. In: Proceedings of IEEE congress on evolutionary computation, pp 1666–1670, Honolulu, HI, USA, May 10–14, 2002

Huang T, Mohan AS (2005) A hybrid boundary condition for robust particle swarm optimization. Antennas Wirel Propag Lett 4:112–117

Ide A, Yasuda K (2005) A basic study of adaptive particle swarm optimization. Electr Eng Jpn 151(3):41–49

Ivatloo BM (2013) Combined heat and power economic dispatch problem solution using particle swarm optimization with time varying acceleration coefficients. Electr Power Syst Res 95(1):9–18

Jamian JJ, Mustafa MW, Mokhlis H (2015) Optimal multiple distributed generation output through rank evolutionary particle swarm optimization. Neurocomputing 152:190–198

Jia D, Zheng G, Qu B, Khan MK (2011) A hybrid particle swarm optimization algorithm for high-dimensional problems. Comput Ind Eng 61:1117–1122

Jian W, Xue Y, Qian J (2004) An improved particle swarm optimization algorithm with neighborhoods topologies. In: Proceedings of 2004 international conference on machine learning and cybernetics, pp 2332–2337, Shanghai, China, August 26–29, 2004

Jiang CW, Bompard E (2005) A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization. Math Comput Simul 68:57–65

Jie J, Zeng J, Han C (2006) Adaptive particle swarm optimization with feedback control of diversity. In: Proceedings of 2006 international conference on intelligent computing (ICIC2006), pp 81–92, Kunming, China, August 16–19, 2006

Jin Y, Cheng H, Yan J (2005) Local optimum embranchment based convergence guarantee particle swarm optimization and its application in transmission network planning. In: Proceedings of 2005 IEEE/PES transmission and distribution conference and exhibition: Asia and Pacific, pp 1–6, Dalian, China, Aug 15–18, 2005

Juang YT, Tung SL, Chiu HC (2011) Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions. Inf Sci 181:4539–4549

Kadirkamanathan V, Selvarajah K, Fleming PJ (2006) Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Trans Evolut Comput 10(3):245–255

Kennedy J (1997) Minds and cultures: particle swarm implications. In: Proceedings of the AAAI Fall 1997 symposium on communicative action in humans and machines, pp 67–72, Cambridge, MA, USA, Nov 8–10, 1997

Kennedy J (1998) The behavior of particle. In: Proceedings of the 7th annual conference on evolutionary program, pp 581–589, San Diego, CA, Mar 10–13, 1998

Kennedy J (1999) Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In: Proceedings of the IEEE international conference on evolutionary computation, pp 1931–1938, San Diego, CA, Mar 10–13

Kennedy J (2000) Stereotyping: Improving particle swarm performance with cluster analysis. In: Proceedings of the IEEE international conference on evolutionary computation, pp 303–308

Kennedy J (2003) Bare bones particle swarms. In: Proceedings of the 2003 IEEE swarm intelligence symposium (SIS'03), pp 80–87, Indianapolis, IN, USA, April 24–26, 2003

Kennedy J (2004) Probability and dynamics in the particle swarm. In: Proceedings of the IEEE international conference on evolutionary computation, pp 340–347, Washington, DC, USA, July 6–9, 2004

Kennedy J (2005) Why does it need velocity? In: Proceedings of the IEEE swarm intelligence symposium (SIS'05), pp 38–44, Pasadena, CA, USA, June 8–10, 2005

Kennedy J, Eberhart RC (1995) Particle swarm optimization? In: Proceedings of the IEEE international conference on neural networks, pp 1942–1948, Perth, Australia

Kennedy J, Mendes R (2002) Population structure and particle swarm performance. In: Proceedings of the IEEE international conference on evolutionary computation, pp 1671–1676, Honolulu, HI, USA, Sept 22–25, 2002

Kennedy J, Mendes R (2003) Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms. In: Proceedings of the 2003 IEEE international workshop on soft computing in industrial applications (SMCia/03), pp 45–50, Binghamton, New York, USA, Oct 12–14, 2003

Krink T, Lovbjerg M (2002) The life cycle model: combining particle swarm optimisation, genetic algorithms and hillclimbers. In: Lecture notes in computer science (LNCS) No. 2439: proceedings of parallel problem solving from nature VII (PPSN 2002), pp 621–630, Granada, Spain, 7–11 Dec 2002

Lee S, Soak S, Oh S, Pedrycz W, Jeonb M (2008) Modified binary particle swarm optimization. Prog Nat Sci 18:1161–1166

Lei K, Wang F, Qiu Y (2005) An adaptive inertia weight strategy for particle swarm optimizer. In: Proceedings of the third international conference on mechatronics and information technology, pp 51–55, Chongqing, China, Sept 21–24, 2005

Leontitsis A, Kontogiorgos D, Pagge J (2006) Repel the swarm to the optimum. Appl Math Comput 173(1):265–272

Li X (2004) Better spread and convergence: particle swarm multiobjective optimization using the maximin fitness function. In: Proceedings of genetic and evolutionary computation conference (GECCO2004), pp 117–128, Seattle, WA, USA, June 26–30, 2004

Li X (2010) Niching without niching parameters: particle swarm optimization using a ring topology. IEEE Trans Evolut Comput 14(1):150–169

Li X, Dam KH (2003) Comparing particle swarms for tracking extrema in dynamic environments. In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC'03), pp 1772–1779, Canberra, Australia, Dec 8–12, 2003

Li Z, Wang W, Yan Y, Li Z (2011) PS-ABC: a hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems. Expert Syst Appl 42:8881–8895

Li C, Yang S, Nguyen TT (2012) A self-learning particle swarm optimizer for global optimization problems. IEEE Trans Syst Man Cybernet Part B Cybernet 42(3):627–646

Li Y, Zhan Z, Lin S, Zhang J, Luo X (2015a) Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems. Inf Sci 293:370–382

Li Z, Nguyena TT, Chen S, Khac Truong T (2015b) A hybrid algorithm based on particle swarm and chemical reaction optimization for multi-object problems. Appl Soft Comput 35:525–540

Liang JJ, Suganthan PN (2005) Dynamic multi-swarm particle swarm optimizer. In: Proceedings of IEEE swarm intelligence symposium, pp 124–129, Pasadena, CA, USA, June 8–10, 2005

Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evolut Comput 10(3):281–295

Lim W, Isa NAM (2014) Particle swarm optimization with adaptive time-varying topology connectivity. Appl Soft Comput 24:623–642

Lim W, Isa NAM (2015) Adaptive division of labor particle swarm optimization. Expert Syst Appl 42:5887–5903

Lin Q, Li J, Du Z, Chen J, Ming Z (2006a) A novel multi-objective particle swarm optimization with multiple search strategies. Eur J Oper Res 247:732–744

Lin X, Li A, Chen B (2006b) Scheduling optimization of mixed model assembly lines with hybrid particle swarm optimization algorithm. Ind Eng Manag 11(1):53–57

Liu Y, Qin Z, Xu Z (2004) Using relaxation velocity update strategy to improve particle swarm optimization. Proceedings of third international conference on machine learning and cybernetics, pp 2469–2472, Shanghai, China, August 26–29, 2004

Liu F, Zhou J, Fang R (2005) An improved particle swarm optimization and its application in longterm stream ow forecast. In: Proceedings of 2005 international conference on machine learning and cybernetics, pp 2913–2918, Guangzhou, China, August 18–21, 2005

Liu H, Yang G, Song G (2014) MIMO radar array synthesis using QPSO with normal distributed contraction-expansion factor. Procedia Eng 15:2449–2453

Liu T, Jiao L, Ma W, Ma J, Shang R (2016) A new quantum-behaved particle swarm optimization based on cultural evolution mechanism for multiobjective problems. Knowl Based Syst 101:90–99

Lovbjerg M, Krink T (2002) Extending particle swarm optimizers with self-organized criticality. In: Proceedings of IEEE congress on evolutionary computation (CEC 2002), pp 1588–1593, Honolulu, HI, USA, May 7–11, 2002

Lovbjerg M, Rasmussen TK, Krink T (2001) Hybrid particle swarm optimizer with breeding and subpopulations. In: Proceedings of third genetic and evolutionary computation conference (GECCO-2001), pp 469–476, San Francisco-Silicon Valley, CA, USA, July 7–11, 2001

Lu J, Hu H, Bai Y (2015a) Generalized radial basis function neural network based on an improved dynamic particle swarm optimization and adaboost algorithm. Neurocomputing 152:305–315

Lu Y, Zeng N, Liu Y, Zhang Z (2015b) A hybrid wavelet neural network and switching particle swarm optimization algorithm for face direction recognition. Neurocomputing 155:219–244

Medasani S, Owechko Y (2005) Possibilistic particle swarms for optimization. In: Applications of neural networks and machine learning in image processing IX vol 5673, pp 82–89

Mendes R, Kennedy J, Neves J (2004) The fully informed particle swarm: simpler maybe better. IEEE Trans Evolut Comput 8(3):204–210

Meng A, Li Z, Yin H, Chen S, Guo Z (2015) Accelerating particle swarm optimization using crisscross search. Inf Sci 329:52–72

Mikki S, Kishk A (2005) Improved particle swarm optimization technique using hard boundary conditions. Microw Opt Technol Lett 46(5):422–426

Mohais AS, Mendes R, Ward C (2005) Neighborhood re-structuring in particle swarm optimization. In: Proceedings of Australian conference on artificial intelligence, pp 776–785, Sydney, Australia, Dec 5–9, 2005

Monson CK, Seppi KD (2004) The Kalman swarm: a new approach to particle motion in swarm optimization. In: Proceedings of genetic and evolutionary computation conference (GECCO2004), pp 140–150, Seattle, WA, USA, June 26–30, 2004

Monson CK, Seppi KD (2005) Bayesian optimization models for particle swarms. In: Proceedings of genetic and evolutionary computation conference (GECCO2005), pp 193–200, Washington, DC, USA, June 25–29, 2005

Mostaghim S, Teich J (2003) Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO). In: Proceedings of the 2003 IEEE swarm intelligence symposium (SIS'03), pp 26–33, Indianapolis, Indiana, USA, April 24–26, 2003

Mu B, Wen S, Yuan S, Li H (2015) PPSO: PCA based particle swarm optimization for solving conditional nonlinear optimal perturbation. Comput Geosci 83:65–71

Netjinda N, Achalakul T, Sirinaovakul B (2015) Particle swarm optimization inspired by starling flock behavior. Appl Soft Comput 35:411–422

Ngoa TT, Sadollahb A, Kima JH (2016) A cooperative particle swarm optimizer with stochastic movements for computationally expensive numerical optimization problems. J Comput Sci 13:68–82

Nickabadi AA, Ebadzadeh MM, Safabakhsh R (2011) A novel particle swarm optimization algorithm with adaptive inertia weight. Appl Soft Comput 11:3658–3670

Niu B, Zhu Y, He X (2005) Multi-population cooperative particle swarm optimization. In: Proceedings of advances in artificial life—the eighth European conference (ECAL 2005), pp 874–883, Canterbury, UK, Sept 5–9, 2005

Noel MM, Jannett TC (2004) Simulation of a new hybrid particle swarm optimization algorithm. In: Proceedings of the thirty-sixth IEEE Southeastern symposium on system theory, pp 150–153, Atlanta, Georgia, USA, March 14–16, 2004

Ozcan E, Mohan CK (1998) Analysis of a simple particle swarm optimization system. In: Intelligent engineering systems through artificial neural networks, pp 253–258

Pampara G, Franken N, Engelbrecht AP (2005) Combining particle swarm optimization with angle modulation to solve binary problems. In: Proceedings of the 2005 IEEE congress on evolutionary computation, pp 89–96, Edinburgh, UK, Sept 2–4, 2005

Park JB, Jeong YW, Shin JR, Lee KY (2010) An improved particle swarm optimization for nonconvex economic dispatch problems. IEEE Trans Power Syst 25(1):156–166

Parsopoulos KE, Vrahatis MN (2002a) Initializing the particle swarm optimizer using the nonlinear simplex method. WSEAS Press, Rome

Parsopoulos KE, Vrahatis MN (2002b) Recent approaches to global optimization problems through particle swarm optimization. Nat Comput 1:235–306

Parsopoulos KE, Vrahatis MN (2004) On the computation of all global minimizers through particle swarm optimization. IEEE Trans Evolut Comput 8(3):211–224

Peer E, van den Bergh F, Engelbrecht AP (2003) Using neighborhoods with the guaranteed convergence PSO. In: Proceedings of IEEE swarm intelligence symposium (SIS2003), pp 235–242, Indianapolis, IN, USA, April 24–26, 2003

Peng CC, Chen CH (2015) Compensatory neural fuzzy network with symbiotic particle swarm optimization for temperature control. Appl Math Model 39:383–395

Peram T, Veeramachaneni k, Mohan CK (2003) Fitness-distance-ratio based particle swarm optimization. In: Proceedings of 2003 IEEE swarm intelligence symposium, pp 174–181, Indianapolis, Indiana, USA, April 24–26, 2003

Poli R (2008) Dynamics and stability of the sampling distribution of particle swarm optimisers via moment analysis. J Artif Evol Appl 10–34:2008

Poli R (2009) Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. IEEE Trans Evolut Comput 13(4):712–721

Poli R, Kennedy J, Blackwell T (2007) Particle swarm optimization—an overview. Swarm Intell 1(1):33–57

Qian X, Cao M, Su Z, Chen J (2012) A hybrid particle swarm optimization (PSO)-simplex algorithm for damage identification of delaminated beams. Math Probl Eng 1–11:2012

Qin Z, Yu F, Shi Z (2006) Adaptive inertia weight particle swarm optimization. In: Proceedings of the genetic and evolutionary computation conference, pp 450–459, Zakopane, Poland, June 25–29, 2006

Ratnaweera A, Halgamuge S, Watson H (2004) Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evolut Comput 8(3):240–255

Reynolds CW (1987) Flocks, herds, and schools: a distributed behavioral model. Comput Graph 21(4):25–34

Richards M, Ventura D (2004) Choosing a starting configuration for particle swarm optimization. In: Proceedings of 2004 IEEE international joint conference on neural networks, pp 2309–2312, Budapest, Hungary, July 25–29, 2004

Richer TJ, Blackwell TM (2006) The levy particle swarm. In: Proceedings of the IEEE congress on evolutionary computation, pp 808–815, Vancouver, BC, Canada, July 16–21, 2006

Riget J, Vesterstrom JS (2002) A diversity-guided particle swarm optimizer—the ARPSO.Technical Report 2002-02, Department of Computer Science, Aarhus University, Aarhus, Denmark

Robinson J, Rahmat-Samii Y (2004) Particle swarm optimization in electromagnetics. IEEE Trans Antennas Propag 52(2):397–407

Robinson J, Sinton S, Rahmat-Samii Y (2002) Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In: Proceedings of 2002 IEEE international symposium on antennas propagation, pp 31–317, San Antonio, Texas, USA, June 16–21, 2002

Roy R, Ghoshal SP (2008) A novel crazy swarm optimized economic load dispatch for various types of cost functions. Electr Power Energy Syst 30:242–253

Salehian S, Subraminiam SK (2015) Unequal clustering by improved particle swarm optimization in wireless sensor network. Procedia Comput Sci 62:403–409

Samuel GG, Rajan CCA (2015) Hybrid: particle swarm optimization-genetic algorithm and particle swarm optimization-shuffled frog leaping algorithm for long-term generator maintenance scheduling. Electr Power Energy Syst 65:432–442

Schaffer JD (1985) Multi objective optimization with vector evaluated genetic algorithms. In: Proceedings of the IEEE international conference on genetic algorithm, pp 93–100, Pittsburgh, Pennsylvania, USA

Schoeman IL, Engelbrecht AP (2005) A parallel vector-based particle swarm optimizer. In: Proceedings of the international conference on neural networks and genetic algorithms (ICANNGA 2005), pp 268–271, Protugal

Schutte JF, Groenwold AA (2005) A study of global optimization using particle swarms. J Glob Optim 31:93–108

Selleri S, Mussetta M, Pirinoli P (2006) Some insight over new variations of the particle swarm optimization method. IEEE Antennas Wirel Propag Lett 5(1):235–238

Selvakumar AI, Thanushkodi K (2009) Optimization using civilized swarm: solution to economic dispatch with multiple minima. Electr Power Syst Res 79:8–16

Seo JH, Im CH, Heo CG (2006) Multimodal function optimization based on particle swarm optimization. IEEE Trans Magn 42(4):1095–1098

Sharifi A, Kordestani JK, Mahdaviania M, Meybodi MR (2015) A novel hybrid adaptive collaborative approach based on particle swarm

optimization and local search for dynamic optimization problems. Appl Soft Comput 32:432–448
Shelokar PS, Siarry P, Jayaraman VK, Kulkarni BD (2007) Particle swarm and ant colony algorithms hybridized for improved continuous optimization. Appl Math Comput 188:129–142
Shi Y, Eberhart RC (1998) A modified particle swarm optimizer. In: Proceedings of the IEEE international conference on evolutionary computation, pp 69–73, Anchorage, Alaska, USA, May 4–9, 1998
Shi Y, Eberhart RC (2001) Fuzzy adaptive particle swarm optimization. In: Proceedings of the congress on evolutionary computation, pp 101–106, IEEE Service Center, Seoul, Korea, May 27–30, 2001
Shin Y, Kita E (2014) Search performance improvement of particle swarm optimization by second best particle information. Appl Math Comput 246:346–354
Shirkhani R, Jazayeri-Rad H, Hashemi SJ (2014) Modeling of a solid oxide fuel cell power plant using an ensemble of neural networks based on a combination of the adaptive particle swarm optimization and levenberg marquardt algorithms. J Nat Gas Sci Eng 21:1171–1183
Sierra MR, Coello CAC (2005) Improving pso-based multi-objective optimization using crowding, mutation and epsilon-dominance. Lect Notes Comput Sci 3410:505–519
Soleimani H, Kannan G (2015) A hybrid particle swarm optimization and genetic algorithm for closedloop supply chain network design in large-scale networks. Appl Math Model 39:3990–4012
Stacey A, Jancic M, Grundy I (2003) Particle swarm optimization with mutation. In: Proceedings of IEEE congress on evolutionary computation 2003 (CEC 2003), pp 1425–1430, Canberra, Australia, December 8–12, 2003
Suganthan PN (1999) Particle swarm optimizer with neighborhood operator. In: Proceedings of the Congress on Evolutionary Computation, pp 1958–1962, Washington, D.C. USA, July 6–9, 1999
Sun J, Feng B, Xu W (2004) Particle swarm optimization with particles having quantum behavior. In: Proceedings of the congress on evolutionary computation, pp 325–331, Portland, OR, USA, June 19–23, 2004
Tang Y, Wang Z, Fang J (2011) Feedback learning particle swarm optimization. Appl Soft Comput 11:4713–4725
Tanweer MR, Suresh S, Sundararajan N (2016) Dynamic mentoring and self-regulation based particle swarm optimization algorithm for solving complex real-world optimization problems. Inf Sci 326:1–24
Tatsumi K, Ibuki T, Tanino T (2013) A chaotic particle swarm optimization exploiting a virtual quartic objective function based on the personal and global best solutions. Appl Math Comput 219(17):8991–9011
Tatsumi K, Ibuki T, Tanino T (2015) Particle swarm optimization with stochastic selection of perturbation-based chaotic updating system. Appl Math Comput 269:904–929
Ting T, Rao MVC, Loo CK (2003) A new class of operators to accelerate particle swarm optimization. In: Proceedings of IEEE congress on evolutionary computation 2003(CEC2003), pp 2406–2410, Canberra, Australia, Dec 8–12, 2003
Trelea IC (2003) The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf Process Lett 85(6):317–325
Tsafarakis S, Saridakis C, Baltas G, Matsatsinis N (2013) Hybrid particle swarm optimization with mutation for optimizing industrial product lines: an application to a mixed solution space considering both discrete and continuous design variables. Ind Market Manage 42(4):496–506
van den Bergh F (2001) An analysis of particle swarm optimizers. Ph.D. dissertation, University of Pretoria, Pretoria, South Africa
van den Bergh F, Engelbrecht AP (2002) A new locally convergent particle swarm optimizer. In: Proceedings of IEEE conference on system, man and cybernetics, pp 96–101, Hammamet, Tunisia, October, 2002
van den Bergh F, Engelbrecht AP (2004) A cooperative approach to particle swarm optimization. IEEE Trans Evolut Comput 8(3):225–239
van den Bergh F, Engelbrecht AP (2006) A study of particle swarm optimization particle trajectories. Inf Sci 176:937–971
Vitorino LN, Ribeiro SF, Bastos-Filho CJA (2015) A mechanism based on artificial bee colony to generate diversity in particle swarm optimization. Neurocomputing 148:39–45
Vlachogiannis JG, Lee KY (2009) Economic load dispatch—a comparative study on heuristic optimization techniques with an improved coordinated aggregation based pso. IEEE Trans Power Syst 24(2):991–1001
Wang W (2012) Research on particle swarm optimization algorithm and its application. Southwest Jiaotong University, Doctor Degree Dissertation, pp 36–37
Wang Q, Wang Z, Wang S (2005) A modified particle swarm optimizer using dynamic inertia weight. China Mech Eng 16(11):945–948
Wang H, Wu Z, Rahnamayan S, Liu Y, Ventresca M (2011) Enhancing particle swarm optimization using generalized opposition-based learning. Inf Sci 181:4699–4714
Wang H, Sun H, Li C, Rahnamayan S, Pan J (2013) Diversity enhanced particle swarm optimization with neighborhood search. Inf Sci 223:119–135
Wen W, Liu G (2005) Swarm double-tabu search. In: First international conference on intelligent computing, pp 1231–1234, Changsha, China, August 23–26, 2005
Wolpert DH, Macready WG (1997) Free lunch theorems for optimization. IEEE Trans Evolut Comput 1(1):67–82
Xie X, Zhang W, Yang Z (2002) A dissipative particle swarm optimization. In: Proceedings of IEEE congression on evolutionary computation, pp 1456–1461, Honolulu, HI, USA, May, 2002
Xie X, Zhang W, Bi D (2004) Optimizing semiconductor devices by self-organizing particle swarm. In: Proceedings of congress on evolutionary computation (CEC2004), pp 2017–2022, Portland, Oregon, USA, June 19–23, 2004
Yang C, Simon D (2005) A new particle swarm optimization technique. In: Proceedings of 17th international conference on systems engineering (ICSEng 2005), pp 164–169, Las Vegas, Nevada, USA, Aug 16–18, 2005
Yang Z, Wang F (2006) An analysis of roulette selection in early particle swarm optimizing. In: Proceedings of the 1st international symposium on systems and control in aerospace and astronautics, (ISSCAA 2006), pp 960–970, Harbin, China, Jan 19–21, 2006
Yang X, Yuan J, Yuan J, Mao H (2007) A modified particle swarm optimizer with dynamic adaptation. Appl Math Comput 189:1205–1213
Yang C, Gao W, Liu N, Song C (2015) Low-discrepancy sequence initialized particle swarm optimization algorithm with high-order nonlinear time-varying inertia weight. Appl Soft Comput 29:386–394
Yasuda K, Ide A, Iwasaki N (2003) Adaptive particle swarm optimization. In: Proceedings of IEEE international conference on systems, man and cybernetics, pp 1554–1559, Washington, DC, USA, October 5–8, 2003
Yasuda K, Iwasaki N (2004) Adaptive particle swarm optimization using velocity information of swarm. In: Proceedings of IEEE international conference on systems, man and cybernetics, pp 3475–3481, Hague, Netherlands, October 10–13, 2004
Yu H, Zhang L, Chen D, Song X, Hu S (2005) Estimation of model parameters using composite particle swarm optimization. J Chem Eng Chin Univ 19(5):675–680
Yuan Y, Ji B, Yuan X, Huang Y (2015) Lockage scheduling of three gorges-gezhouba dams by hybrid of chaotic particle swarm opti-

mization and heuristic-adjusted strategies. Appl Math Comput 270:74–89

Zeng J, Cui Z, Wang L (2005) A differential evolutionary particle swarm optimization with controller. In: Proceedings of the first international conference on intelligent computing (ICIC 2005), pp 467–476, Hefei, China, Aug 23–25, 2005

Zhai S, Jiang T (2015) A new sense-through-foliage target recognition method based on hybrid differential evolution and self-adaptive particle swarm optimization-based support vector machine. Neurocomputing 149:573–584

Zhan Z, Zhang J, Li Y, Chung HH (2009) Adaptive particle swarm optimization. IEEE Trans Syst Man Cybernet Part B Cybernet 39(6):1362–1381

Zhan Z, Zhang J, Li Y, Shi Y (2011) Orthogonal learning particle swarm optimization. IEEE Trans Evolut Comput 15(6):832–847

Zhang L, Yu H, Hu S (2003) A new approach to improve particle swarm optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO 2003), pp 134–139, Chicago, IL, USA, July 12–16, 2003

Zhang R, Zhou J, Moa L, Ouyanga S, Liao X (2013) Economic environmental dispatch using an enhanced multi-objective cultural algorithm. Electr Power Syst Res 99:18–29

Zhang L, Tang Y, Hua C, Guan X (2015) A new particle swarm optimization algorithm with adaptive inertia weight based on bayesian techniques. Appl Soft Comput 28:138–149