INVITED PAPER *Special Section on Large Scale Algorithms for Learning and Optimization*

Particle Swarm Optimization – A Survey

Keisuke KAMEYAMA[†a)], ***Member***

SUMMARY Particle Swarm Optimization (PSO) is a search method which utilizes a set of agents that move through the search space to find the global minimum of an objective function. The trajectory of each particle is determined by a simple rule incorporating the current particle velocity and exploration histories of the particle and its neighbors. Since its introduction by Kennedy and Eberhart in 1995, PSO has attracted many researchers due to its search efficiency even for a high dimensional objective function with multiple local optima. The dynamics of PSO search has been investigated and numerous variants for improvements have been proposed. This paper reviews the progress of PSO research so far, and the recent achievements for application to large-scale optimization problems.

key words: *particle swarm optimization, swarm intelligence*

1. Introduction

This paper serves as a survey on a group of optimization algorithms known as Particle Swarm Optimization (PSO). PSO was originally introduced by Kennedy and Eberhart in 1995 [1]. After its introduction, the simplicity and the flexibility of the algorithm achieving a very efficient search of near-optimal solutions even for problems with quite a tricky search space landscape, has attracted many researchers who try to analyze and improve it, and also those who intend to apply it.

The original research by Kennedy and Eberhart started out to acquire local agent rules for simulating the social behavior of a flock of birds or a school of fish. Inspired by the works such as [2] by Reynolds, who programmed a flock-simulating group of agents (*boids*) by sole description of agent interaction rules, they started to try various interaction rules among the agents. Soon, they found that some type of interaction in the flock can benefit the efficiency of finding an important location, such as food in the flock's domain of activity. They saw this nature effective as a novel algorithm for general nonlinear programming, so they improved and simplified the rule for updating the agent's position, making use of some local memory and information sharing. Thus the Particle Swarm Optimization was born.

Since then, various modifications to the original PSO have been proposed. In parallel, they are being applied to various fields that require parameter optimization in a high-dimensional space, favored due to the algorithm's simplicity and its high efficiency in searching. This survey will try to follow the progress of PSO research.

In the following, the original PSO will be reviewed and the trends of research in analyzing and improving the original algorithm in various aspects will be introduced in Sect. 2. In Sect. 3, works in several major research directions will be reviewed. Section 4 is devoted to current research trends towards using the PSO in large-scale optimization problems. Finally, in Sect. 5, the paper will be closed with concluding remarks.

2. Particle Swarm Optimization (PSO)

2.1 PSO in Its Original Form

Particle Swarm Optimization (PSO) is a method for finding the global minimum of a scalar valued objective function $f(\boldsymbol{x}) \in \boldsymbol{R}$ defined on domain $\boldsymbol{D} = \boldsymbol{R}^n$. The global minimum will be referred to as $\boldsymbol{x}_0 \in \boldsymbol{R}^n$ in the following. Thus $f(\boldsymbol{x}_0) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \boldsymbol{R}^n$. All the following discussions can be applied to cases when the global maximum is sought, by multiplying -1 to the objective function f.

In PSO, a group of N agents search through the domain $\boldsymbol{D}$. Each agent called the *particle* will be denoted by c_i ($i = 1, \ldots, N$), and the whole set of particles is referred to as the *swarm* $\boldsymbol{S} = \{c_1, \ldots, c_N\}$. In Fig. 1, a schematic of the search

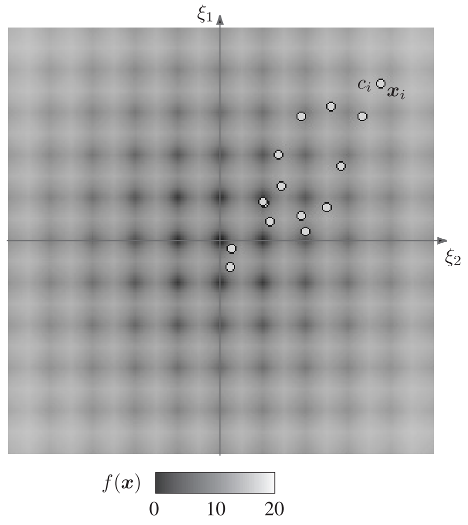

Fig. 1 Particle Swarm in search of the global minimum $\boldsymbol{x}_0 = (\xi_1, \xi_2)^T = (0, 0)^T$ of the Rastrigin function defined on a 2 dimensional domain. White circle indicate the position ($\boldsymbol{x}_i$) of each particle (c_i).

Manuscript received December 19, 2008.
Manuscript revised March 8, 2009.

[†]The author is with the Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba-shi, 305–8573 Japan.
a) E-mail: Keisuke.Kameyama@cs.tsukuba.ac.jp

DOI: 10.1587/transinf.E92.D.1354

by the swarm is shown for an objective function defined on a 2 dimensional domain.

Each particle in the swarm moves through the domain $\boldsymbol{D}$ in search of $\boldsymbol{x}_0$ where function f will take the global minimum. The searching trajectory of each particle is determined by a simple law incorporating its individual observation, memory and information shared among the particles in its *neighborhood.* The neighborhood $\boldsymbol{S}_i$ of particle c_i is a given subset of the swarm ($\boldsymbol{S}_i \in \boldsymbol{S}$), usually defined in the proximity of c_i on the particle's indexing space. The definition of neighborhood will be discussed in detail in Sects. 2.2.2 and 3.2.

Particle c_i holds its current coordinate in the search space $\boldsymbol{x}_i$ and its velocity $\boldsymbol{v}_i$. The fitness of $\boldsymbol{x}_i$ is evaluated with the objective function f and will be fed back to the particle as $f(\boldsymbol{x}_i)$. Each particle also holds its maximally evaluated coordinate so far, as $\boldsymbol{p}_i$. The point at which the particles in the neighborhood $\boldsymbol{S}_i$ scored best so far is transmitted to c_i as $\boldsymbol{g}_i$. Vectors $\boldsymbol{p}_i$ and $\boldsymbol{g}_i$ are often called the *gbest* and *pbest* vectors, standing for the global best and the particle's best findings, respectively.

The particles move through the search space according to the updating rules of their coordinates and velocity defined as,

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1), \tag{1}$$

$$\begin{aligned}\boldsymbol{v}_i(t+1) = \boldsymbol{v}_i(t) &+ \phi_1 R(\boldsymbol{p}_i - \boldsymbol{x}_i(t)) \\ &+\phi_2 R(\boldsymbol{g}_i - \boldsymbol{x}_i(t)).\end{aligned} \tag{2}$$

Here, t and R denote the time and a random diagonal matrix $diag(\rho_1, \rho_2, \ldots, \rho_n)$ respectively. Each diagonal element ρ_i is a uniform random number in $[0, 1]$ which is re-generated upon every evaluation of R. There are also cases in which a single random scalar $\rho \in [0, 1]$ is used instead of matrix R. Parameters $\phi_1 \geq 0$ and $\phi_2 \geq 0$ are to be determined in beforehand.

As seen in Eq. (2), the velocity $\boldsymbol{v}_i$ is generally updated towards $\boldsymbol{p}_i$ and $\boldsymbol{g}_i$, weighted by parameters ϕ_1 and ϕ_2, with randomness induced by R or ρ. This nature will tend the particle search to the direction where good solutions were found in the past.

The parameters ϕ_1 and ϕ_2 determine the tendency of velocity update. A large ϕ_1 will shift the velocity towards the particle's best $\boldsymbol{p}_i$. Similarly, a large ϕ_2 will direct the particle towards the neighbor group's best $\boldsymbol{g}_i$. In effect, particle c_i will take an uneven *swarming* movement around

$$\frac{1}{\phi_1 + \phi_2}(\phi_1 \boldsymbol{p}_i + \phi_2 \boldsymbol{g}_i). \tag{3}$$

Recommended values of the parameters in the original paper [1] was $\phi_1 = \phi_2 = 2$ to have the particle fly over the bests by a 1/2 chance.

2.2 Analysis, Issues and Research Directions

After its initial introduction, several issues residing in the original PSO have been pointed out by various researchers.

2.2.1 Stability and Convergence

One significant issue found out by many of those who have applied PSO was its instability. Often, the trajectory of each particle becomes a multidimensional oscillation of a limited amplitude [3]. Under some parameter settings, the particle velocity monotonically increases, and results in a so-called *explosion* of the swarm [4].

Even when the swarm becomes concentrated around a small region including a local minimum of f, final convergence to the local minimum tended to be rare, or took a very long time.

2.2.2 Neighborhood

Another aspect of PSO that has been investigated is the selection of each particle's neighborhood $\boldsymbol{S}_i$.

The simplest setup for the particle neighborhood is to include all particles in the neighborhood; namely, $\boldsymbol{S}_i = \boldsymbol{S}$, $(i = 1, \ldots, N)$. In this case, the particles have the tightest communication and the findings of the whole swarm is immediately shared by every particle in the swarm as the common *gbest* vector $\boldsymbol{g} = \boldsymbol{g}_1 = \cdots = \boldsymbol{g}_N$. The neighborhood of this type, is called the *gbest-type*.

Another extremity of neighborhood definition would be the total isolation, in which particles do not exchange *pbest* information at all. However, it has been proved that the swarm of this type (*pbest-type*) performs very poorly.

Intermediate approaches where a particle shares information with a subset $\boldsymbol{S}_i \in \boldsymbol{S}$ is known to perform better, generally. This setup is called the *lbest-type* in which the shared best $\boldsymbol{g}_i$ is the best among the experiences of particles in $\boldsymbol{S}_i$; this shared vector is often called the *lbest* which stands for local-best [5].

3. Modifications and Analysis

In accordance with the issues summarized in the previous section, various modifications to the original PSO have been proposed so far. Also, novel ideas from other disciplines such as evolutionary algorithms have been imported to the framework of PSO. Several among those directions will be reviewed in this Section.

3.1 Stability and Convergence

3.1.1 Limiting Particle Speed

Under some settings, the swarm can become scattered in a very wide area in the search domain, resulting in a very high particle speed. In order to avoid such an *explosion*, a practice to limit the maximum particle speed component as v_{max} has been used. The velocity update rule with this saturation is given by

$$v'_{id}(t+1) = \begin{cases} v_{id}(t+1) & (v_{id}(t+1) \leq v_{max}) \\ v_{max} & (v_{id}(t+1) > v_{max}) \end{cases}$$

$$(d = 1, \dots, n), \tag{4}$$

where $\boldsymbol{v}_i(t) = [v_{i1}(t) \cdots v_{in}(t)]^T$ [1]. Here, $\boldsymbol{v}'_i(t+1) = [v'_{i1}(t+1) \cdots v'_{in}(t+1)]^T$ is the modified velocity.

Alternatively,

$$\boldsymbol{v}'_i(t+1) = \begin{cases} \boldsymbol{v}_i(t+1) & (v \le v_{max}) \\ \frac{v_{max}}{v}\boldsymbol{v}_i(t+1) & (v > v_{max}) \end{cases} \tag{5}$$

with $v = \|\boldsymbol{v}_i(t+1)\|$, may be used to maintain the velocity direction.

Use of v_{max} often contributes to stabilizing the swarm and improves the chance that the swarm will be lead to the neighborhood of $\boldsymbol{x}_0$. However, convergence of the particles to $\boldsymbol{x}_0$ is not guaranteed. Often it takes a long time for one of the particles just happen to visit $\boldsymbol{x}'_0 \simeq \boldsymbol{x}_0$ giving a near-optimal reading $f(\boldsymbol{x}'_0)$.

3.1.2 Inertia Weight

Another attempt to improve the stability of the search is the use of *inertia weight* w, which controls the amount the present velocity affects the velocity of the next time step. This method was introduced by Shi and Eberhart [6]. Using the inertia weight, the rule in Eqs. (1) and (2) become

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1), \tag{6}$$

$$\begin{aligned} \boldsymbol{v}_i(t+1) = \ & w\boldsymbol{v}_i(t) + \phi_1 R(\boldsymbol{p}_i - \boldsymbol{x}_i(t)) \\ & + \phi_2 R(\boldsymbol{g}_i - \boldsymbol{x}_i(t)). \end{aligned} \tag{7}$$

Here, a large w will make it relatively hard to change the particle's movement direction. As a result, the swarm will be scattered in a large portion of the search domain $\boldsymbol{D}$. This nature is desirable in the initial phase of the search when the promising portion in $\boldsymbol{D}$ is still unknown.

As the initial exploration reveals the coarse landscape of $f(\boldsymbol{x})$, the search should be focused to smaller promising regions for a finer search. In this phase, a smaller inertia weight is more suitable. Scheduling the optimization to start with $w = 1$, and to gradually reduce it (e.g. exponentially) so that it tends to $w = 0$, is known to be a good practice.

3.1.3 Constriction

Clerc and Kennedy [4] investigated the PSO as a class of dynamic systems that can be controlled by choosing the coefficient parameters in the system equation. They started investigating the nature of PSO as a multidimensional version of the dynamic system governed by equations

$$v(t+1) = v(t) + \phi y(t) \tag{8}$$

$$y(t+1) = -v(t) + (1-\phi)y(t). \tag{9}$$

By assuming

$$y(t) = \frac{\phi_1 p + \phi_2 g}{\phi_1 + \phi_2} - x(t) \tag{10}$$

$$\phi = \phi_1 + \phi_2, \tag{11}$$

it is found that the system of Eqs. (8) and (9) is equivalent to the 1-dimensional version of the PSO system defined in Eqs. (1) and (2) with randomness omitted.

By putting

$$\boldsymbol{P}(t) = \begin{bmatrix} v(t) \\ y(t) \end{bmatrix} \tag{12}$$

and

$$M = \begin{bmatrix} 1 & \phi \\ -1 & 1-\phi \end{bmatrix}, \tag{13}$$

the system can be described as $\boldsymbol{P}(t+1) = M\boldsymbol{P}(t)$, whose dynamics will be determined by matrix M. Through an eigenvalue analysis of M, the authors found connections between the value of ϕ and some significant dynamics of PSO. Among them are the cyclic trajectories of the particles and the divergence of particle velocity.

Further on, they discussed that the original PSO system in the form of Eqs. (8) and (9) is a special case of a general system

$$v(t+1) = \alpha v(t) + \beta\phi y(t) \tag{14}$$

$$y(t+1) = -\gamma v(t) + (\delta - \eta\phi)y(t), \tag{15}$$

where $\phi > 0$ and coefficients $\{\alpha, \beta, \gamma, \delta, \eta\}$ further modify the dynamics of the system. The authors classify the systems by the relations among the five coefficients, and show that velocity explosion and convergence can be controlled by the selection of these coefficients. Note that the system with inertia weight corresponds to the special case of $\alpha = \gamma = w$ and $\beta = \delta = \eta = 1$.

Among such generalized PSO models, the *Class 1"* system which has constraints in the coefficients as $\alpha = \beta = \gamma = \eta$ and $\delta = 1$, will have a system equation of

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1), \tag{16}$$

$$\begin{aligned} \boldsymbol{v}_i(t+1) = \ & \chi(\boldsymbol{v}_i(t) + \phi_1 R(\boldsymbol{p}_i - \boldsymbol{x}_i(t)) \\ & + \phi_2 R(\boldsymbol{g}_i - \boldsymbol{x}_i(t))). \end{aligned} \tag{17}$$

In this system, the convergence of the swarm to a local minimum is guaranteed, by choosing the *constriction* coefficient χ as,

$$\chi = \begin{cases} \frac{2\kappa}{|\phi - 2 + \sqrt{\phi^2 - 4\phi}|} & (\phi > 4) \\ \kappa & (otherwise) \end{cases} \tag{18}$$

where parameter $\kappa \in (0,1)$, slows the convergence as it approaches 1.0.

In the experiments in [4], optimization by various methods were compared on some well known benchmark objective functions such as the De Jong's set [5], [7], Schaffer's f6, Griewank, Rosenbrock and Rastrigin functions [5]. Benchmark objective functions commonly used in evaluation of evolutionary and swarm optimization algorithms are listed in Table 1.

In their results, it was shown that appropriate use of constriction parameters in the generalized classes of PSO achieves convergence to the global minimum at high rate. In

Table 1 Benchmark functions commonly used in evolutionary and swarm optimization research.

DeJong's f1(Sphere)	$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^n$	$\boldsymbol{x} \in \boldsymbol{R}^n$
DeJong's f2	$f(\boldsymbol{x}) = 100(x_1^2 - x_2^2)^2 + (1 - x_1)^2$	$\boldsymbol{x} \in \boldsymbol{R}^2$
DeJong's f3	$f(\boldsymbol{x}) = \sum_{i=1}^{n} \lfloor x_i \rfloor$	$\boldsymbol{x} \in \boldsymbol{R}^n$
DeJong's f4	$f(\boldsymbol{x}) = \sum_{i=1}^{n} i \cdot x_i^4 + Gauss(0,1)$	$\boldsymbol{x} \in \boldsymbol{R}^n$
DeJong's f5(Foxholes)	$f(\boldsymbol{x}) = \frac{1}{0.002} + \sum_{i=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})}$	$\boldsymbol{x} \in \boldsymbol{R}^2, a_{ij} \in [-1, 1]$
Schaffer's f6	$f(\boldsymbol{x}) = 0.5 + \frac{(sin\sqrt{x_1^2 + x_2^2}) - 0.5}{(1.0 + 0.001(x_1^2 + x_2^2))^2}$	$\boldsymbol{x} \in \boldsymbol{R}^2$
Griewank	$f(\boldsymbol{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$\boldsymbol{x} \in \boldsymbol{R}^n$
Rosenbrock	$f(\boldsymbol{x}) = \sum_{i=1}^{n-1}\{100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\}$	$\boldsymbol{x} \in \boldsymbol{R}^n$
Rastrigin	$f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2 + 10 - 10\cos(2\pi x_i)$	$\boldsymbol{x} \in \boldsymbol{R}^n$

contrast, it showed that sole use of speed truncation (v_{max}) was not sufficient. However, it was also shown that the joint use of both techniques can further improve the search and convergence within a certain number of update iterations.

3.1.4 Stability in a Stochastic Particle System

The analysis that lead to a now-common practice of using the constriction parameter in [4] assumed a deterministic system with no randomness. A recent work by Kadirkamanathan et al. [8] analyzes the stability of PSO with stochastic particle dynamics including the scalar randomness (ρ) in Eq. (7), by way of Lyapunov stability analysis. They derived the condition of asymptotic stability as,

$$K < \frac{2(1 - 2|w| + w^2)}{1 + w} \quad (|w| < 1,\ w \neq 0) \tag{19}$$

where $0 < \phi < K$ holds for ϕ in Eq. (11). This result gives a guide for selecting the parameters ϕ and w for stability and convergence.

3.2 Neighborhood

Improvements to the original PSO in determining each particle's neighborhood $\boldsymbol{S}_i$ will be reviewed here.

3.2.1 Index Topologies

In the lbest-type PSO, neighborhood of particle c_i is often a set of particles in the vicinity of c_i in the particle indexing structure. In this case, the neighborhoods will have some overlaps. Therefore, the lbest information ($\boldsymbol{g}_i$ and $\boldsymbol{g}_j$) of different particles (c_i and c_j) will be mutually affected via the pbest $\boldsymbol{p}_k$ of particle c_k which is included in both neighborhoods ($\boldsymbol{S}_i$ and $\boldsymbol{S}_j$). In Fig. 2 (a), an example of an indexing

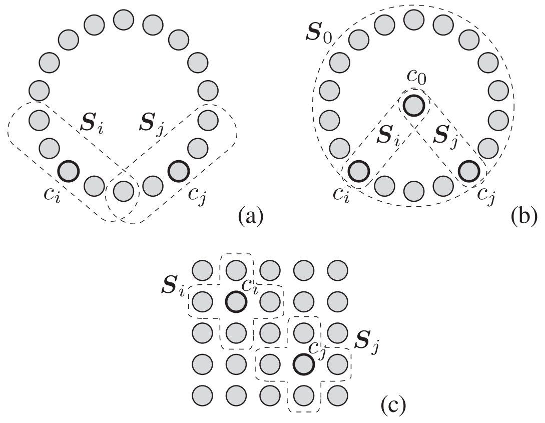

Fig. 2 Examples of neighborhood definitions employed by the *lbest* type PSO (in dashed lines). (a) Circular : Particles in immediate $k = 2$ neighborhood in a circular index are set as the neighborhood $\boldsymbol{S}_i$ of each particle c_i. (b) Wheel : The swarm has particle c_0 which acts as the channel for information sharing. Other particles have neighborhoods of limited subsets always including particle c_0. (c) von Neumann : Four immediate neighbors in the particles indexed in a grid on a torus surface comprise the neighborhood.

structure with circular topology is shown where $\boldsymbol{S}_i$ includes other particles within arc distance k.

Another method of lbest implementation is shown in Fig. 2 (b). It is the *wheel* topology which uses a *focus* particle c_0. The focus particle is special in that $\boldsymbol{S}_0 = \boldsymbol{S}$. The neighborhood of other particles always include the focus. In this configuration, the focus c_0 gradually moves towards the global best found so far by the swarm. By its movement towards the global best, the focus c_0 will always indicate the "global trend" of the search by way of its pbest $\boldsymbol{p}_o$.

When compared with the gbest-type PSO, lbest-type swarm tends to be slow in convergence, but is less likely to be captured in a local minimum because *remote* neighborhoods (on the indexing structure) tend to search in a different candidate regions [9]. In effect, this is equivalent to the maintenance of the diversity which plays an important role in avoiding early convergence of evolutionary algorithms.

In the experiments for several functions in Table 1, Kennedy and Mendes tested various neighborhood types on a Class 1" PSO [9]. The types included gbest, lbests with $k = 1$ on a circular topology, $k = 1$ on a 3D-pyramid topology, a wheel in Fig. 2 (b), von Neumann topology in Fig. 2 (c), and a swarm divided into several isolated neighborhood subsets. The findings from the experiments were as follow.

- von Neumann lbest generally achieved best in convergence speed and solution quality.
- 3D-pyramid topology scored second best.
- Isolated subsets and wheel topology gave moderate results.
- Circular lbest was slowest to converge with worse-than-moderate solutions.
- gbest had the worst performance, being quick to converge, mostly to one of the local minima.
- Results were problem dependent. No all-around superior neighborhood topology was found.

Other works on neighborhoods based on indexing topologies include Lane et al. [10] finding Delauney triangular grids effective in low dimensional search domains.

3.2.2 Locality in Search Space

In contrast to these static neighborhoods based on predefined index topology, Suganthan [11] proposed to dynamically redefine them using the immediate particle positions in the search space $\boldsymbol{D}$ while running the PSO. The neighborhood $\boldsymbol{S}_i$ of particle c_i was redefined as a set of particles inside a hypersphere in $\boldsymbol{D}$ centered at $\boldsymbol{x}_i$. In addition, the radius of the hypersphere will gradually grow, eventually becoming a gbest-type PSO. Experimental results show that this method found better solutions in average, when it was compared with the conventional gbest-type PSO. Binkley and Hagiwara [12] also used locality in the search space by augmenting the gbest factor in Eq. (17) with a function of immediate particle distances. They also found improvements over the methods with index-based neighborhoods on some benchmark functions.

3.3 Introduction of Genetic Operations

Ideas common in other evolutionary optimization algorithms have been imported to PSO, and are known as *hybridization* approaches.

3.3.1 Selection

Angeline [13] introduced a variant of PSO which additionally incorporates the selection mechanism. A form of tournament selection evaluating the current particle fitness is performed, and all the particles will be sorted according to their scores. Then the positions and the velocities of the better half of the swarm are copied to the inferior half, but maintaining each particle's pbest vectors for particle diversity. By this operation, the focus of the inferior half will be turned to a more promising subset of the search space. In the experiments, the PSO with the original dynamics was compared with the same PSO with additional selection and significant improvements are reported for some test objective functions.

3.3.2 Mutation

Mutation operations common in real-valued genetic algorithms (GA) have been introduced to PSO as well. Injection of randomness in the positional updates of particles have been done using Gaussian mutation (Higashi and Iba [14]) and Cauchy mutation (Stacy et al. [15]). Improvements for both benchmark objective functions and real-world problems have been reported. Someya [16] reports that the search tendencies (exploratory vs. exploiting) could be controlled by modifying an asymmetric normal distribution for the density of randomness for the diagonal elements of $\boldsymbol{R}$ in the last term of Eq. (2).

3.3.3 Diversity

Maintenance of diversity is also important in finding a good solution in a high-dimensional search domain. Delayed information propagation by sparsely connected neighborhoods contributes in this aspect. Other modifications specially aimed for the maintenance of diversity include particle replacement from a dense cluster (Løvbjerg and Krink [17]), use of repulsion force to avoid particle collision (Krink et al. [18]), and occasional switching of dynamics among the sub-swarms (Al-kazemi and Mohan [19]). Stochastic switching of particle dynamics with a similar aim is found in Vesterstrøm et al. [20].

See Olorunda and Engelbrecht [21] for various definitions of swarm diversity and their features.

3.3.4 Speciation

Introduction of species to the particles, namely *speciation* is another technique for maintaining the particle diversity. Iwamatsu [22] proposed the multi-species PSO (MSPSO), and independently, Parrot and Li [23] proposed the species-based PSO (SPSO). In these models, several fittest particles in the swarm will become the *species seeds* and will include other particles in the neighborhood to its species. Additionally, the seed particle will act as the lbest of each species neighborhood. In both MSPSO and SPSO, isolation of species and gradual migration of particles upon re-speciation contributed to efficient search for the global optimum of multimodal objective functions. Also, SPSO has been modified for tracking multiple optima of a dynamically changing multimodal objective functions [23].

3.4 Generalized Velocity Update Rules

In the original PSO, particle velocity update is only dependent to the particle's pbest ($\boldsymbol{p}_i$) and gbest ($\boldsymbol{g}_i$) as in Eq. (2). Mendes et al. [24] extended this selective dependence to all the particles in the neighborhood; namely to make each particle *fully informed* of the pbest of the particles in $\boldsymbol{S}_i$. The modified update rule with constriction χ is,

$$\boldsymbol{v}_i(t+1) = \chi(\boldsymbol{v}_i(t) + \phi(\boldsymbol{q}_i - \boldsymbol{x}_i(t))) \tag{20}$$

where $\boldsymbol{q}_i$ is a weighted sum of $\boldsymbol{p}_j$ for all particles in $\boldsymbol{S}_i$. The weights are set according to various aspects of the particles. Such aspects include the evaluation of each particle's pbest ($f(\boldsymbol{p}_j)$) and the distance to particle c_i ($\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$). Liang et al. [25] use a similar update rule as Eq. (20), with each element of $\boldsymbol{q}_i$ being chosen from the elements of $\boldsymbol{p}_j$ ($c_j \in \boldsymbol{S}_i$) via a tournament selection.

3.5 Multiobjective Optimization

Modifications to PSO for solving multiobjective optimization problems (MOP) have been introduced by several authors. The objective in this case is to find the Pareto front in the search space, as a set of points satisfying the Pareto optimality.

In [26], Hu et al. applies PSO to MOPs by employing a dynamic neighborhood scheme (as in [11]), with sequential optimization of objective functions, thereby finding the Pareto front.

Coello Coello et al. [27] proposed the multiobjective PSO (MOPSO) which keeps a repository of Pareto optimal points in the search space, and uses one among them as the gbest vector ($\boldsymbol{g}_i$) in the velocity updating of each particle. In addition, a mutation operation to avoid the local minima is used. Mutation is also applied to the system parameters (such as w, ϕ_1 and ϕ_2 in Eq. (7)), for an extensive exploration.

3.6 Multimodal Functions and Global Optimization

Although PSO is quite robust in escaping the local minima, there is no guarantee that the swarm will find the global minimum. Naturally the search becomes harder in multimodal objective functions. Parsopulos and Vrahatis [28] introduced a sequential modification to the objective function in the neighborhood of each local minimum found. The particles are additionally repelled from these local minima in order that the swarm will eventually find the global minimum.

3.7 Manipulation of the Search Space

Voss [29] proposed a dual-PSO system, in which two sets of PSO runs in a localized subspace in $\boldsymbol{D}$ derived by principal component analysis (PCA) of past pbest and gbest vector collection, and in $\boldsymbol{D}$ itself. It is based on a heuristics that a parallel search in a promising subspace can contribute to an efficient search. In the experiments, improvements in the search efficiency is reported.

3.8 Optimization in a Dynamic Environment

When the objective function changes dynamically, the PSO system needs to detect and adapt to the changed environment. Again, maintenance of diversity in the swarm is considered to be the key. Hu and Eberhart [30] propose to occasionally re-evaluate the objective function at the gbest found so far. Upon detecting the change of the objective function, re-randomization of particle positions and erasing the pbest information have been proposed for re-triggering exploration in a wide area.

Blackwell and Branke [31] use the *multiswarm* which consists of multiple subswarms designated to converge to different local extrema. Diversity among the subswarms is maintained by *exclusion* which re-initializes a subswarm converged at an extremum already taken by another subswarm, and *anti-convergence* which re-initializes one of the subswarms when there are no more exploratory subswarms. Further, the diversity within the subswarm is maintained by a modeled Coulomb repulsion among particles to prevent the convergence of the subswarms. Also, as previously mentioned, the dynamic version of SPSO (DSPSO) by Parrot and Li [23] also successfully tracks multiple moving optima.

4. Towards Applications to Large-Scale Optimization Problems

4.1 Improving Exploration and Exploitation

Although numerous early works report PSO's high efficiency in finding a near-optimal solution in small-scale problems ($n \approx 30$ or less), PSO does suffer from the curse of dimensionality in large-scale problems as in other optimization methods employing parallel search [32]. The issue here is how to balance *exploration* in the vast search space and *exploitation* at the proximity of promising regions, using a limited number of particles. Most of the genetic operations reviewed in Sec. 3.3 contribute to exploration which enforces a divide-and-conquer strategy in some way.

Division of the swarm into subswarms, or the multiswarm approach [31] are among such approaches. In Dynamic Multi-Swarm PSO (DMSPSO) by Liang and Suganthan [33], random re-definition of small-sized (3-5 particles) neighborhood occurs, thereby forcing exchange of information among the particles. Recently, the modified DMSPSO with exploitation by Quasi-Newton method by Zhao et al. [34]. achieved comparable to better results to other optimization algorithms in the large-scale optimization competition held at the CEC2008 conference.

As an example of other additional exploitation features, Maeda [35] introduces PSO with local search using simultaneous perturbation. The method is reported to outperform PSO with constriction in convergence efficiency for several problems.

As another approach of task division, Van den Bergh and Engelbrecht [36] introduced Cooperative PSO (CPSO) which divides the swarm to multiple subswarms assigned to searches in particular low-dimensional subspaces. The communication among the subswarms are achieved by sharing the best-achieving components upon evaluating the objective function.

4.2 Parallel Implementation

As the problem scale grows, increasing the number of particles will certainly improve the chance of finding a good solution. Several authors have pursued parallel implementation of PSO.

Schutte et al. [37] implemented PSO in parallel by dividing the swarm population to processor nodes. In their setup, only the evaluation of the objective function f was done in parallel in a synchronized manner. Experimental results using a cluster computer up to 32 nodes showed good scalability provided that objective functions require near-identical computational costs for evaluation.

Li and Wada [38] also assigned a subswarm to each processor node running in parallel. In order to solve the bottleneck for speed up in exchanging information between the nodes, they proposed the *delayed exchange* scheduling which can omit global synchronization. It is reported that, by this improvement, an improved scalability was achieved in a PC cluster.

In McNabb et al. [39], PSO was coded on a parallel computing framework called MapReduce, and was evaluated using a cluster with max. 128 nodes. Similar scalability as former works were observed up to a certain number of nodes. However, using more nodes, it is reported that implementation and communication overheads dominated and hindered further improvements.

5. Conclusion

This paper reviewed the progress of research on Particle Swarm Optimization (PSO). It is obvious that only a tiny portion of the literature could be referred to, and application papers had to be totally omitted. Research on PSO, to improve its performance, to modify it for various objectives, and to apply it to various problems, is so divergent (an *explosion*, so to say). Over 1600 articles can be found in IEEE publications alone (http://ieeexplore.ieee.org), over 450 articles are found in CiNii database which reflects research activity in Japan (http://ci.nii.ac.jp/en), and more than 177000 web pages in relation with PSO exist throughout the world, according to Google (http://www.google.com). These numbers reflect the ever-increasing interest of researchers and engineers in PSO.

References

[1] J. Kennedy and R.C. Eberhart, "Particle swarm optimization," IEEE International Conf. on Neural Networks, pp.1942–1948, 1995.

[2] C.W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," Comput. Graph., vol.21, no.4, pp.25–34, 1987.

[3] E. Ozcan and C.K. Mohan, "Particle swarm optimization: Surfing the waves," Congress on Evolutionary Computation, pp.6–9, 1999.

[4] M. Clerc and J. Kennedy, "The particle swarm – Explosion, stability, and convergence in a multidimensional complex space," IEEE Trans. Evol. Comput., vol.6, no.1, pp.58–73, 2002.

[5] J. Kennedy and R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann, 2001.

[6] Y. Shi and R.C. Eberhart, "A modified particle swarm optimizer," IEEE International Conf. on Evolutionary Computation, pp.69–73, 1998.

[7] K.A. DeJong, An analysis of the behavior of a class of genetic adaptive systems, Doctoral Dissertation, University of Michigan, Ann Arbor, 1975.

[8] V. Kadirkamanathan, K. Selvarajah, and P.J. Fleming, "Stability analysis of the particle dynamics in particle swarm optimizer," IEEE Trans. Evol. Comput., vol.10, no.3, pp.245–255, 2006.

[9] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," Congress on Evolutionary Computation, pp.1671–1676, 2002.

[10] J. Lane, A. Engelbrecht, and J. Gain, "Particle swarm optimization with spatially meaningful neighbours," IEEE Swarm Intelligence Symposium, 2008.

[11] P.N. Suganthan, "Particle swarm optimiser with neighbourhood operator," Congress on Evolutionary Computation, pp.1958–1962, 1999.

[12] K.J. Binkley and M. Hagiwara, "Particle swarm optimization with area of influence: Increasing the effectiveness of the swarm," IEEE Swarm Intelligence Symposium, pp.45–52, 2005.

[13] P.J. Angeline, "Using selection to improve particle swarm optimization," IEEE International Conf. on Evolutionary Computation, pp.84–89, 1998.

[14] N. Higashi and H. Iba, "Particle swarm optimization with gaussian mutation," IEEE Swarm Intelligence Symposium, pp.72–79, 2003.

[15] A. Stacey, M. Jancic, and I. Grundy, "Particle swarm optimization with mutation," Congress on Evolutionary Computation, pp.1425–1430, 2003.

[16] H. Someya, "Cautious particle swarm," IEEE Swarm Intelligence Symposium, 2008.

[17] M. Løvbjerg and T. Krink, "Extending particle swarm optimisers with self-organized criticality," Congress on Evolutionary Computation, pp.1588–1593, 2002.

[18] T. Krink, J.S. Vesterstrøm, and J. Riget, "Particle swarm optimisation with spatial particle extension," Congress on Evolutionary Computation, pp.1474–1479, 2002.

[19] B. Al-kazemi and C.K. Mohan, "Multi-phase generalization of the particle swarm optimization algorithm," Congress on Evolutionary Computation, pp.489–494, 2002.

[20] J.S.Vesterstrøm, J. Riget, and T. Krink, "Division of labor in particle swarm optimisation," Congress on Evolutionary Computation, pp.1570–1575, 2002.

[21] O. Olorunda and A.P. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," IEEE Congress on Evolutionary Computation, pp.1128–1134, 2008.

[22] M. Iwamatsu, "Locating all the global minima using multi-species particle swarm optimizer: The inertia weight and the constriction factor variants," IEEE Congress on Evolutionary Computation, pp.816–822, 2006.

[23] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," IEEE Trans. Evol. Comput., vol.10, no.4, pp.440–458, 2006.

[24] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," IEEE Trans. Evol. Comput., vol.8, no.3, pp.204–210, 2004.

[25] J.J. Liang, A.K. Qin, P.N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of

multimodal functions," IEEE Trans. Evol. Comput., vol.10, no.3, pp.281–295, 2006.

[26] X. Hu, R.C. Eberhart, and Y. Shi, "Particle swarm with extended memory for multiobjective optimization," IEEE Swarm Intelligence Symposium, pp.193–197, 2003.

[27] C.A. Coello Coello, G.T. Pulido, and M.S. Lechuga, "Handling multiple objectives with particle swarm optimization," IEEE Trans. Evol. Comput., vol.8, no.3, pp.256–279, 2004.

[28] K.E. Parsopulos and M.N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," IEEE Trans. Evol. Comput., vol.8, no.3, pp.211–224, 2004.

[29] M.S. Voss, "Principal component particle swarm optimization (PCPSO)," IEEE Swarm Intelligence Symposium, pp.401–404, 2005.

[30] X. Hu and R.C. Eberhart, "Adaptive particle swarm optimization: Detection and response to dynamic systems," Congress on Evolutionary Comp., pp.1666–1670, 2002.

[31] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," IEEE Trans. Evol. Comput., vol.10, no.4, pp.459–472, 2006.

[32] S. Piccand, M. O'Neill, and J. Walker, "On the scalability of particle swarm optimisation," IEEE Congress on Evolutionary Computation, pp.2505–2512, 2008.

[33] J.J. Liang and P.N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," IEEE Swarm Intelligence Symposium, pp.124–129, 2005.

[34] S.Z. Zhao, J.J. Liang, P.N. Suganthan, and M.F. Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization," IEEE Congress on Evolutionary Comp., pp.3845–3852, 2008.

[35] Y. Maeda and T. Kuratani, "Simultaneous perturbation particle swarm optimization," IEEE Congress on Evolutionary Computation, pp.672–676, 2006.

[36] F.v. Bergh and A.P. Engelbrecht, "A cooperative approach to particle swarm optimization," IEEE Trans. Evol. Comput., vol.8, no.3, pp.225–239, 2004.

[37] J.F. Schutte, J.A. Reinbolt, B.J. Fregly, R.T. Haftka, and A.D. George, "Parallel global optimization with the particle swarm algorithm," Int. J. Numerical Methods in Engineering, vol.61, no.13, pp.2296–2315, 2004.

[38] B. Li and K. Wada, "Parallelizing particle swarm optimization," Pacific Rim Conf. on Comm., Computers and Signal Pro., pp.288–291, 2005.

[39] A.W. McNabb, C.K. Monson, and K.D. Seppi, "Parallel PSO using Mapreduce," IEEE Congress on Evolutionary Computation, pp.7–14, 2007.

Keisuke Kameyama received the B.E., M.E., and Dr. Eng. degrees in electrical engineering (1989), precision machinery engineering (1991), and computer science (1999), respectively, all from Tokyo Institute of Technology, Tokyo, Japan. From 1992 to 2000 he worked as a Research Associate at Tokyo Institute of Technology. In 2000, he joined the Tsukuba Advanced Research Alliance (TARA), University of Tsukuba. Since 2007, he works as an Associate Professor at the Graduate School of Systems and Information Engineering, University of Tsukuba. His research interests include pattern recognition, signal processing, and adaptive information processing methods. Dr. Kameyama is a member of IEEE and AVIRG.