Scatter PSO – A More Effective Form of Particle Swarm Optimization

Peng-Yeng Yin, Fred Glover, Manuel Laguna, and Jia-Xian Zhu

***Abstract*—A fertile complementarity exists between scatter search (SS) and particle swarm optimization (PSO). Shared and contrasting principles underlying these methods provide a fertile basis for combining them to create a hybrid method. We identify a specific hybrid, Scatter PSO, giving rise to two variants that prove more effective than the constriction factor model of PSO. Applied to finding global minima for continuous nonlinear functions, Scatter PSO not only is able to obtain better solutions to a widely used set of benchmark functions, but also proves more robust under a variety of experimental conditions.**

I. Introduction

PARTICLE swarm optimization (PSO) [1] has emerged as an effective solver for complex optimization problems, demonstrating successful outcomes in many applications, including evolving weights and structure for artificial neural networks [2], CNC end milling [3], reactive power and voltage control [4], state estimation for electric power distribution systems [5], and curve segmentation [6], just to name a few. These successes are not due to highly complex or sophisticated implementation of the PSO approach; in fact the PSO systems underlying these applications can be coded in just a few lines.

Several variations of PSO have been developed but most of them resemble the standard PSO form. Recently, Mendes et al. [7] proposed a PSO system that informs each particle of the set of best outcomes obtained in the process of examining every other particle in its neighborhood, rather than just the single best outcome found by examining its neighbors, which has been shown to improve the chances of finding better solutions. Such a strategy resembles a more general theme embodied in scatter search, where a current solution (particle) can profit from information derived from all others in a common *reference set*, not restricted simply to neighbors of the solution.

More broadly, PSO has several features spiritually similar to those found in the Scatter Search and Path Relinking (SS/PR) Template [8]. The SS/PR template dynamically maintains and updates a reference set of the best solutions observed in history and uses subsets of the reference set to generate new solutions. These new solutions are then improved by heuristic means and compared with current members of the reference set for its continued updating. The combined solutions can be viewed as points resulting from relinking different search paths by treating one of the source solutions as an *initial point* and the others as *guiding points*. (A good overview of the interrelated scatter search and path relinking methods, including a compilation of applications, appears in [9].)

PSO can be compared to the SS/PR template by viewing the particle as an initial point and the previous best outcomes (solutions) derived from this particle and its neighbors as the guiding points, which thereby are used to change the particle's position. There are, however, some substantial differences between the two approaches. For example, PSO keeps track of the previous best solution of each particle and that of its neighbors, while the SS/PR template maintains a reference set of the best solutions that have been observed overall in history and does not keep track of the individual's trajectory. (An exception occurs in the common situation where scatter search or path relinking is combined with tabu search, which exploits the trajectory of the individual using adaptive memory strategies.) A more conspicuous difference is that PSO constrains each particle to change its position by applying one trial of interaction with the best information, whereas the SS/PR template provides a basis for combining multiple solutions from subsets of the reference set.

This paper investigates several elements derived by considering the foregoing relationships that profoundly affect PSO performance, as embodied the following issues expressed in terms of the swarm metaphor. (1) What information existing in the swarm is more informative and essential to enriching the swarm intelligence? (2) How do we utilize this information to influence an individual particle's behavior and improve the overall system performance? Several alternatives for marrying the points of view of PSO and the SS/PR template exist, and we propose specific instances leading to the Scatter PSO algorithm, whose encouraging computational results suggest preliminary answers to the above-mentioned questions.

The remainder of this paper is organized as follows. Section 2 reviews the standard PSO and its variations. In Section 3, we propose two new PSO extensions with several new features. Section 4 presents the experimental results and discussions. Finally, a conclusion is made in Section 5.

P. Y. Yin is with the Department of Information Management, National Chi Nan University, 1 University Rd., Puli, Nantou 545, Taiwan (phone: 886-49-2910960; fax: 886-49-2915205; e-mail: pyyin@ncnu.edu.tw).

F. Glover is with the Leeds School of Business, University of Colorado, Boulder, CO 80309 USA, and also with OptTek Systems, Inc., 1919 Seventh Street, Boulder, CO 80302 USA (e-mail: Fred.Glover@colorado.edu).

M. Laguna is with the Leeds School of Business, University of Colorado, Boulder, CO 80309 USA (e-mail: laguna@colorado.edu).

J. X. Zhu is with the Department of Information Management, National Chi Nan University, 1 University Rd., Puli, Nantou 545, Taiwan (e-mail: s95213527@ncnu.edu.tw).

1-4244-1340-0/07$25.00 ©2007 IEEE

II. VARIATIONS OF THE PARTICLE SWARM ALGORITHMS

The theory on which the PSO is founded is *sociocognition*, which states that social cognition happens in the interactions among individuals in such a manner that each individual learns from its neighbors' models/patterns, especially from those learning experiences that are rewarded. Cognition emerges from the convergence of individuals' beliefs. PSO is also biologically inspired, drawing on the observation that any member in a swarm will try to move as close to the center of the swarm as possible because an individual at the edge of the swarm is more likely to be caught by a predator. This form of swarm intelligence not only reduces the probability of being attacked but also increases the chance of success for food foraging. (We note that this latter aspect of the swarm intelligence metaphor, in spite of being instrumental in publicizing swarm methods, is a bit questionable and even unnecessary. Clearly, its assumptions merely require that *some* individuals have a proclivity to seek central positions. Moreover, in some situations the health and survivability of the swarm may be enhanced by the presence of *competent* individuals that seek positions nearer the swarm boundary, where they may better defend the swarm against impending threats.)

The PSO algorithm facilitates simple rules for changing the velocities of moving particles in a swarm and has been shown to be an effective solver to optimization problems. Kennedy and Eberhart [1][10] had developed both of the binary- and continuous-encoding variable versions of the PSO. In this paper, we focus on continuous PSO, which is more commonly used. The standard continuous PSO algorithm is summarized in Fig. 1.

1. Initialize.
   1.1 Generate N particle positions, $\vec{P}_i = (p_{i1}, p_{i2}, \ldots, p_{ir})$, $1 \le i \le N$, at random.
   1.2 Generate N velocity vectors, $\vec{V}_i = (v_{i1}, v_{i2}, \ldots, v_{ir})$, $1 \le i \le N$, at random.
2. Repeat until a stopping criterion is met.
   2.1 Evaluate the fitness of each particle $\vec{P}_i$.
   2.2 Determine the best vector $pbest_i$ visited so far by each particle $\vec{P}_i$.
   2.3 Determine the best vector *nbest* observed so far by the neighbors of particle $\vec{P}_i$.
   2.4 For each particle $\vec{P}_i$, update velocity vector $\vec{V}_i$ by
   $v_{ij} \leftarrow w v_{ij} + \varphi_1 rand_1(pbest_{ij} - p_{ij}) + \varphi_2 rand_2(nbest_j - p_{ij}), \forall j = 1, \ldots, r$
   2.5 For each particle $\vec{P}_i$, update particle's position by
   $p_{ij} \leftarrow p_{ij} + v_{ij}, \forall j = 1, \ldots, r$

Fig. 1. Summary of the standard continuous PSO algorithm.

Given an optimization problem characterized by r real-valued decision variables, standard PSO initiates a swarm of N particles generated at random, each of which is represented as $\vec{P}_i = (p_{i1}, p_{i2}, \ldots, p_{ir})$. Thus, the swarm of particles represents a pool of candidate solutions for the optimization problem. A velocity vector $\vec{V}_i = (v_{i1}, v_{i2}, \ldots, v_{ir})$ is created at random for each particle and is repeatedly updated to guide the search direction for the particle. During each iteration of the main loop (Step 2 in Fig. 1), the *fitness* of each particle is evaluated. Then, the particle's personal best ($pbest_i$) and neighbors' best (*nbest*) are determined. There are at least two versions for defining the neighbors' best position. In the local version, each particle keeps track of the best position *lbest* attained by its neighbors defined in a neighborhood topology. A ring structure is usually used as the neighborhood topology, i.e., each particle is connected to two other particles and the last particle is considered as the next-door neighbor of the first particle. Note that the neighborhood topology is not defined according to the shortest distance in the solution space or fitness space, but rather is somewhat arbitrarily given as a function of the neighbors in the social network that the particle communicates with. For the global version where each individual particle is connected to every other, the best position *gbest* is determined by reference to all particles in the swarm. Hence, the global version is a special case of the local version. To conduct social learning, each particle updates its velocity $\vec{V}_i$ and position $\vec{P}_i$ through each variable dimension j using Eqs. (1) and (2) as follows.

$$v_{ij} \leftarrow w v_{ij} + \varphi_1 rand_1(pbest_{ij} - p_{ij}) + \varphi_2 rand_2(nbest_j - p_{ij}) \quad (1)$$

and

$$p_{ij} \leftarrow p_{ij} + v_{ij} \quad (2)$$

where $w < 1$ is the inertia weight, φ_1 and φ_2 are the cognitive coefficients, and $rand_1$ and $rand_2$ are random real numbers drawn from $U(0, 1)$. Thus, the particle explores a potential region defined by the personal best and the neighbors' best, while the cognitive coefficients and the random multipliers change the weightings for the two bests at every iteration. The inertia weight is analogous to a step size in classical nonlinear programming, and, favors moving the particle in the same direction as it moved at the previous iteration. A dynamically decreasing value of the inertia weight is usually used to focus more on exploration search at early iterations and transit to put more emphasis on exploitation search at later iterations.

Clerc and Kennedy [11] used another scheme called the Type 1″ constriction coefficient to control the convergence of the algorithm by replacing Eq. (1) with the following,

$$v_{ij} \leftarrow K(v_{ij} + \varphi_1 rand_1(pbest_{ij} - p_{ij}) + \varphi_2 rand_2(nbest_j - p_{ij})) \quad (3)$$

and

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \quad (4)$$

where $\varphi = \varphi_1 + \varphi_2$ and must be greater than 4.0. Typically, φ is set to 4.1 and K is thus 0.729.

Mendes et al. [7] pointed out that the constriction

coefficient does not limit the use of φ shared between two terms, it is only necessary that the parts sum to a value that is appropriate for K. It implies that the particle velocity can be adjusted using any number of terms. The authors of [7] have studied a number of weighting schemes to combine all neighbors' information instead of only using the best among them. Let ω_k estimate the relevance of social influence from particle k, the change of velocity $\vec{V}_i$ can be performed by

$$v_{ij} \leftarrow K(v_{ij} + \varphi(mbest_{ij} - p_{ij})) \quad (5)$$

where

$$mbest_{ij} = \frac{\sum_{k \in N} \omega_k \varphi_k pbest_{kj}}{\sum_{k \in N} \omega_k \varphi_k} \text{ and } \varphi_k = U\left[0, \frac{\varphi_{\max}}{N}\right]. \quad (6)$$

Therefore, the particle is fully informed by all its neighbors defined in the social network (neighborhood topology). Mendes' algorithms, called FIPS (Fully Informed Particle Swarm), consistently outperform the standard constriction coefficient PSO on several benchmark functions.

PSO systems have manifested encouraging achievements in many applications by using surprisingly simple formulas. In many of these cases, only the personal best experiences (outcomes) and the neighbors' best experiences are maintained. Our work here is motivated by the conjecture that the performance of the PSO algorithms can be improved by taking advantage of knowledge implicitly available in the swarm by exploiting appropriate ideas from scatter search and path relinking.

III. Proposed Algorithms

We first introduce notation that is useful for presenting our Scatter PSO algorithms, which consist of two novel variants whose features are subsequently described.

A. Notation

We use a triplet 'd/S/Δ' to differentiate among the alternative algorithmic variants we investigate in the process of isolating the specific variants that compose the Scatter PSO approach. In the terminology of the SS/PR template, the first symbol d indicates the number of guiding points used in the algorithm and the second symbol S refers to the identity of the sets of guiding points. The last symbol Δ is a character which can be either '+' or '–', where Δ = '+' indicates that the velocity updating formula (Eq. (3)) of the constriction coefficient is included in the algorithm, and Δ = '–' indicates that this formula is excluded. For example, the algorithm $2/\{(pbest_k, gbest) | \forall k\}/+$ uses two guiding points, $pbest_k$ and $gbest$, for the velocity updating of the ith particle as follows.

$$v_{ij} \leftarrow K(v_{ij} + \varphi_1 rand_1(pbest_{kj} - p_{ij}) + \varphi_2 rand_2(gbest_j - p_{ij})) \quad (7)$$

This formula is applied for each value of k, so we need to keep the multiple trials of v_{ij} plus the one generated by the standard formula (Eq. (3)) and pick the best among them as the final decision for the updated value of v_{ij}. Thus, the algorithm will increase the number of function evaluations per iteration. Finally, the ith particle position is updated using the same formula $p_{ij} \leftarrow p_{ij} + v_{ij}$, $\forall j$.

B. Exploiting Swarm Intelligence

Inspired by the FIPS model and the SS/PR template, we propose two strategies to exploit the knowledge accumulated by the swarm. The first strategy induces social learning in each particle by drawing on the best experience (solution found) in the swarm and the best experience of any other particle. The second strategy uses the members from the reference set as guiding points to determine the direction in which to move the particle. We show that both strategies obtain results superior to those obtained by the constriction coefficient PSO method on the set of benchmark functions tested.

1) Learning From The Swarm Members: Most PSO algorithms model social influence by reference to a single instance of interaction between each particle and its local best neighbor. However, the region found by the best neighbor may not be better than that found by the second best, or the third best, etc. It may be a better idea to retain the interaction with every neighbor instead of only communicating with the best one among them. In contrast to the customary conception, we anticipate that the social influence on an individual particle may be more precisely modeled by a set of interactions between this particle and others in the swarm. To increase the chance of locating a good region, the swarm's best (or the neighbors' best) is involved in every interaction, making use of another notion derived from the SS/PR template. In particular, we propose the algorithm $3/\{(pbest_i, gbest, pbest_k) | \forall k \neq i\}/-$. Hence, there are three guiding points involved in the change of each particle's velocity, namely the particle's own best (i.e., the best found in the course of its trajectory), the swarm's best (the best found in its trajectory of any particle in the swarm), and the best of a given second particle (where each particle is selected in turn to take the role of the second particle). The weighting between the three guiding points can be performed in a number of ways. In this study, we use equal weighting, fitness weighting, and self weighting. The equal weighting gives the same weight for each guiding point, the fitness weighting determines the weight according to the guiding point's fitness, and the self weighting gives the particle's own best half of the total weight and the two other guiding points share the other half of the weight.

Our algorithm is substantially different from the FIPS model which constrains the single interaction between each particle and its neighbors and incorporates a weighted average of the individual bests of all the neighbors in this interaction. Instead, we conduct multiple interactions for every particle. In each interaction, only three carefully selected guiding points are considered. This form of social learning is founded on both theoretical and empirical view points as will be described.

2) Learning from the Reference Set Members: The successes of scatter search and path relinking owe in part to their mechanisms for manipulating and updating the reference set, which incorporates the best solutions observed during the search history. Different subsets of the reference set are selected and the members of each subset are combined to generate new solutions. These new solutions are then improved by heuristic procedures and compared with current members of the reference set for its continued updating. A similar memory structure called the *external archive* has also recently been embodied in the multi-objective PSO (MOPSO) proposed by Coello et al. [12]. The external archive stores the non-dominated solutions observed so far and the MOPSO algorithm randomly chooses one member from this archive to serve in place of the neighbors' best solution. In this respect, the idea of the reference set (or external archive) provides a link between the SS/PR template and developments more recently introduced in PSO,

Our second strategy, unlike the first strategy which enables each particle to learn from every member of the swarm, prescribes that each particle instead learns from interactions with members of the reference set. For this, we maintain an array of historical best solutions $RefSet[i]$, $i = 1, \ldots, R$, where R is the size of the reference set. $RefSet[i]$ is sorted according to the decreasing order of the members' fitness values, i.e., $RefSet[1]$ indicates the best solution in the reference set and $RefSet[R]$ represents the worst of them. In particular, we propose the algorithm $3/\{(pbest_i, RefSet[1], RefSet[k]) | \forall k > 1\}/-$. The three guiding points involved are the particle's own best, the best member and any other member in the reference set. We use the same weighting schemes as those used in the first strategy, namely, the equal weighting, fitness weighting, and self weighting.

3) Features Of Our Algorithms: Our algorithms have a number of distinct features and that afford potential advantages over other PSO-based algorithms. These features are summarized as follows.

- Strategically selected guiding points: PSO researchers and practitioners have historically focused on maximizing the benefit to be derived from best experiences (either personal best or neighbors' best). But there is potentially rich information contained in the swarm that is not limited to these best experiences that may help improve the system performance, if properly exploited. We have conducted extensive experiments to identify what information is essential for enriching the swarm intelligence. We observed that the region explored by using only the best solution for guidance may not be better than the region explored by additionally using the second best solution for guidance (or the third best, etc.) Moreover, interacting with a set of globally ranked best solutions by using the reference set of the SS/PR template proves more beneficial than just using personal best and neighbors' best solutions as in customary PSO approaches.

Our experimental results disclose that $pbest_i$ and *gbest* (or *lbest* in the local version) are essential guiding points for capturing the proper social influence effects. The removal of either term causes the overall performance to significantly deteriorate. From the perspective of tabu search, whose origins overlap with those of scatter search and path relinking, this is due to the fact that the inclusion of $pbest_i$ induces a useful intensification of the search over the particular local region in which the particle lies, while the guidance contributed by *gbest* contributes an element of global intensification that likewise enhances the quality of the solution generated. If both $pbest_i$ and *gbest* are present in the social influence process, the addition of another guiding point, either $pbest_k$ or $RefSet[k]$, adds a balancing diversification effect that succeeds in significantly improving overall performance.

- Dynamic social network: Most PSO algorithms use a *static* social network throughout their execution. Each individual particle always interacts with the same neighbors defined in the neighborhood topology, thus providing what may be viewed as a limited context for transmitting social influence. By contrast, our proposed algorithms incorporate the dynamic perspective of scatter search that enlarges the context of interactions in a manner analogous to eliciting multiple viewpoints as a basis for influencing the individual is by the group it communicates with. Each such viewpoint is provided by an interaction with another individual in the swarm or the reference set, and the learner benefits from the influence of the best of all these interactions. Consequently, the social network is *dynamic*, allowing the particle to determine the best neighborhood topology as a consequence of the influence of the indicated interactions. As a motivation, we observe that the social network in the human world is also dynamic. People may get involved in different social networks at different times due to travel, changing residences, graduation, job-switching, and many other social activities. It is a truism that dynamic social networks can provide more fruitful information sharing than static social networks.
- Parsimony: We choose to use three guiding points in both variants of our Scatter PSO algorithm, which is a number between that used by the standard PSO (which relies on two guiding points) and that used by the FIPS model (which treats all neighbors as guiding points). Our experimental results demonstrate that a navigation guided by more than three points in our algorithms, employing any of the three weighting schemes (equal weighting, fitness weighting and self weighting) spoils the performance if the algorithms are given the same computational resource. We conjecture that this occurs because the information is blurred when incorporating too many terms in the social influence process at the same time. This finding is also supported in previous relevant works. The SS/PR template emphasizes the relevance of combining between 3 and 5 solutions, and the scatter search study of Campos et al. [13] finds that most of the high quality solutions come from combinations using at most 3 reference solutions. So if computational resources are limited, it can be worthwhile to limit attention to using 2 or 3 reference solutions. Mendes et al. [7] also found that their

FIPS algorithms perform best when a smaller neighborhood size of 2 to 4 neighbors is used, and increasing the neighborhood size causes the system performance to deteriorate. However, the approach of [7], as previously noted, uses the same neighborhood topology at every iteration.

IV. Experimental Results And Discussion

Our experimental results from comparing the performance of the proposed Scatter PSO algorithmic variants with that of competing methods disclose several interesting outcomes. The platform for conducting the experiments is a PC with a 1.8 GHz CPU and 1.0 GB RAM. All programs are coded in C++ language.

TABLE I
Benchmark Testing Functions

Function name	Formula	Range of variables	Global optimum solution
Sphere	$\sum_{i=1}^{30} x_i^2$	[-100.0, 100.0]	$x_i = 0.0, \forall i$
Rosenbrock	$\sum_{i=1}^{29}\left(100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right)$	[-30.0, 30.0]	$x_i = 1.0, \forall i$
Rastrigin	$\sum_{i=1}^{30}\left(x_i^2-10\cos\left(2\pi x_i\right)+10\right)$	[-5.12, 5.12]	$x_i = 0.0, \forall i$
Griewank	$\frac{1}{4000}\sum_{i=1}^{30} x_i^2-\prod_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600.0, 600.0]	$x_i = 0.0, \forall i$

We evaluated comparative performance on a set of benchmark functions that have been extensively used in the literature. Table 1 lists the function names, formulas, range of variables, and the known globally optimal solutions. All functions have 30 variables and are minimization functions whose optimum function values are zero. As each function has a different range of function values, we need to scale the results according to a reference method. For instance, it is easy for most algorithms to obtain a result of 0.01 on Sphere, however, a result of 40.0 on Rosenbrock is considered good. Thus, we propose a performance measure, merit = f/f_{ref}, where f is the function value obtained by the testing algorithm and f_{ref} is that obtained by the reference method. As all the functions involve minimization, a smaller value of merit indicates a better performance. Here, we use the Type 1″ constriction coefficient [11] as the reference method. So a testing algorithm is considered to outperform the Type 1″ constriction coefficient if its value of merit is lesser than 1.0, and it becomes worse when the merit is growing greater than 1.0. To make a fair comparison, the same parameter values as those used in the Type 1″ algorithm have been used in all of the tested Scatter PSO algorithmic variants, but are shared among multiple terms if the variant uses more than two guiding points. All competing algorithms use a swarm of 20 particles, and the reference set maintains 10 reference points.

The evaluation criteria for the testing algorithms are as follows. Each algorithm is executed for 100 independent runs on every testing function. The average of the values of merit over these repetitive runs is reported. Repetitive runs are performed to reduce the bias in the reported results that can arise from the randomized nature of the algorithms. For each single run of a given algorithm tested, the stopping criterion is to terminate upon performing 160,000 function evaluations. This value is chosen with the goal of ensuring that performance is evaluated when the tested algorithm has converged.

A. Learning from $pbest_k$

We evaluate the performance of the first variant of Scatter PSO $3/\{(pbest_i, gbest, pbest_k)|\forall k \neq i\}/-$ (which will be referred to as A-equal, A-fitness, and A-Self for the three different weighting schemes, respectively) that allows each particle to learn from every other member in the swarm. We also examine what information is essential when learning from $pbest_k$ and the removal of this information will mar the system performance. First, two sub-variants are examined to analyze the importance of the particle's own best, $pbest_i$. The first sub-variant is $2/\{(pbest_i, pbest_k)\ |\forall k \neq i\}/-$ (referred to as B) which includes $pbest_i$ in every interaction with $pbest_k$. The other sub-variant $2/\{(gbest, pbest_k)|\forall k \neq i\}/-$ (referred to as C), instead uses two guiding points *gbest* and $pbest_k$ for each *k*, but excludes $pbest_i$. Similarly, another two sub-variants are explored to testify the significance of the *gbest*. One is $2/\{(pbest_i, pbest_k)|\forall k \neq i, pbest_k \neq gbest\}/+$ (referred to as D) which embeds the constriction coefficient formula (Eq. (3)) that contains *gbest* as one of the guiding points. The other sub-variant $2/\{(pbest_i, pbest_k)|\forall k \neq i, pbest_k \neq gbest\}/-$ (referred to as E), however, excludes *gbest* from the learning process.

The average merit obtained by these algorithms over 100 repetitive runs on each of the testing functions is shown in Fig. 2. Because the function Sphere is relatively simple to solve compared to the other functions, all of the algorithms (including the reference method) can obtain the globally optimal function value, so they have an average merit of 1.0 on Sphere. For the rest of the functions tested, all the A-type algorithms using three guiding points ($3/\{(pbest_i, gbest, pbest_k)|\forall k \neq i\}/-$) perform significantly better than the reference method. It seems that the A-self procedure

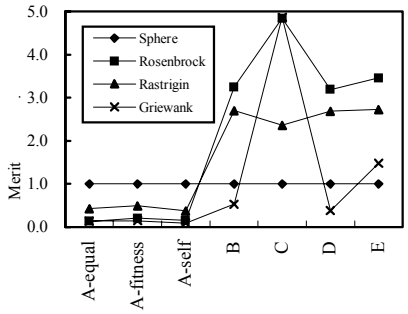

Fig. 2. Comparative performance of algorithms learning from $pbest_k$.

that weights the particle's own previous best by .5 is superior to the other two weighting schemes; however, the difference is not remarkable.

We also see that algorithm B significantly outperforms algorithm C on Rosenbrock and Griewank but it is slightly worse on Rastrigin, indicating that the particle's own previous best is an essential component of information when learning from other members in the swarm. This is because the inclusion of $pbest_i$ induces the particle to exploit its local region while trying to improve its solution quality. Otherwise the particle will tend to wander about in the whole solution space due to the guidance of $pbest_k$ without sufficiently attending to a particular region of interest. Analogously, algorithm D beats algorithm E on all functions except Sphere on which they both obtain the optimum value. So the inclusion of *gbest* in the set of guiding points also helps the particle improve its solution. Our A-type algorithms use $pbest_i$, *gbest*, and $pbest_k$, for each k as the three guiding points. The multiple perspectives influenced by $pbest_k$ provide valuable information which the traditional constriction coefficient (the reference method) lacks.

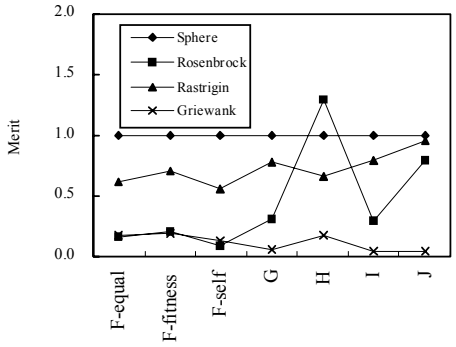

Fig. 3. Comparative performance of algorithms learning from *RefSet*[*k*].

B. Learning from RefSet[k]

In this section, the performance of the proposed second type of Scatter PSO algorithms, 3/{($pbest_i$, *RefSet*[1], *RefSet*[*k*])|$\forall k$>1}/− (whose sub-variants will be referred to as F-equal, F-fitness, and F-Self, respectively) is evaluated for the three different weighting schemes. These F-type algorithms let each particle learn from every member of the reference set. Similarly, the significance of the guidance using $pbest_i$ and *gbest* (which is *RefSet*[1] in this learning scenario) is attested to by the performance of the following competing algorithms. First, the algorithmic sub-variant 2/{($pbest_i$, *RefSet*[*k*])|$\forall$*RefSet*[*k*]≠$pbest_i$}/− (referred to as G) includes $pbest_i$ in the set of guiding points, while the other sub-variant 2/{(*RefSet*[1], *RefSet*[*k*]) | $\forall k$>1 and *RefSet*[*k*] ≠ $pbest_i$}/− (referred to as H) replaces $pbest_i$ by *RefSet*[1]. Second, the sub-variant 2/{($pbest_i$, *RefSet*[*k*]) | $\forall k$>1 and *RefSet*[*k*]≠$pbest_i$}/+ (referred to as I) uses the constriction coefficient formula (Eq. (3)) that contains *gbest* as one of the guiding points, while the other sub-variant 2/{($pbest_i$, *RefSet*[*k*])|$\forall k$>1 and *RefSet*[*k*]≠$pbest_i$}/− (referred to as J) excludes *gbest* in the learning.

The average merit of these algorithms tested on each of the functions is shown in Fig. 3. Phenomena similar to those appearing in Fig. 2 are also observed in this figure. Apart from the Sphere function that all competing algorithms can solve optimally, the F-type algorithms beat the reference method quite substantially. The F-self algorithm with the self weighting scheme performs best. The necessity of including $pbest_i$ in the set of guiding points is confirmed by the fact that algorithm G outperforms algorithm H on Rosenbrock and Griewank but is slightly worse on Rastrigin. We also observe that algorithm I beats algorithm J on Rosenbrock and Rastrigin and they perform almost as well as each other on Griewank. So the inclusion of *gbest* in algorithm I also improves its performance.

C. Overall Comparison

To find the best algorithmic instances that solve all the testing functions well, we make an overall comparison by computing the average merit of every algorithmic sub-variant over all functions. Fig. 4 shows the comparative results which convey the important information that the overall performance of A-type algorithms is significantly better than the reference method but the performance of the other variations (algorithms B-E), which use two guiding points, is significantly worse. On the other hand, all algorithmic sub-variants of the second kind (algorithms F-J) can beat the reference method handily. It seems that the algorithms that employ learning by reference to the $pbest_k$ scenario are more vulnerable than those that employ learning by reference to the *RefSet*[*k*] scenario obtained from the SS/PR template. This observation motivates future investigation that incorporates additional features of the SS/PR template into the PSO method, such as the inclusion of a diversification generator and an explicit improvement method.

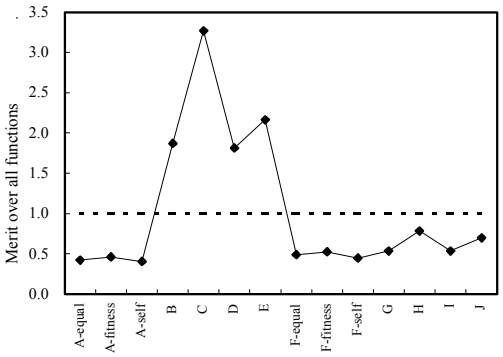

Fig. 4. Overall comparative performance.

To provide a direct view on the improvement in function values, the average function values and the standard deviation over the 100 repetitive runs obtained by the reference method (Type 1″ constriction coefficient) and our principal methods (A-type and F-type) are listed in Table 2. It is seen that our proposed methods are superior to the reference method in producing smaller function values and the results are more consistent over repetitive runs as well.

TABLE II
AVERAGE FUNCTION VALUES AND STANDARD DEVIATIONS

	Sphere	Rosenbrock	Rastrigin	Griewank
Type 1″	0.000 (0.000)	5.936 (9.810)	50.126 (18.028)	0.053 (0.081)
A-equal	0.000 (0.000)	0.757 (1.426)	21.362 (8.183)	0.007 (0.009)
A-fitness	0.000 (0.000)	1.196 (1.827)	24.635 (7.767)	0.008 (0.009)
A-self	0.000 (0.000)	0.887 (1.685)	18.542 (9.863)	0.005 (0.007)
F-equal	0.000 (0.000)	0.997 (1.726)	30.854 (12.155)	0.009 (0.010)
F-fitness	0.000 (0.000)	1.196 (1.827)	35.261 (11.977)	0.009 (0.011)
F-self	0.000 (0.000)	0.559 (1.384)	27.928 (9.164)	0.007 (0.009)

As the A-type and F-type algorithms using three guiding points all perform better than their variations with 2 guiding points, we are interested in investigating the performance of the extended algorithms using four guiding points under both learning criteria, and in finding the answer to whether the addition of the fourth guiding point yields advantages. We extend the A-type algorithms to $4/\{(pbest_i, gbest, pbest_k, pbest_h) | \forall k \neq h \neq i, pbest_k \neq gbest, pbest_h \neq gbest\}/-$, so both of the essential information $pbest_i$ and $gbest$ are included in the set of guiding points and the other two guiding points use every two distinct particles' bests. Because it is no use to put identical points in the set of guiding points (whose learning can be also achieved by using a smaller number of distinct guiding points), we check for duplicate guiding points and avoid these cases to make the algorithm more efficient. The three weighting schemes used by the A-type algorithms can also be applied for the version with four guiding points. Similarly, for the learning with $RefSet[k]$ scenario, we also extend the F-type algorithms to $4/\{(pbest_i, RefSet[1], RefSet[k], RefSet[h]) | \forall k > h > 0\}/-$, so the particle's own best and the swarm's global best are included in the set of guiding points and the other two guiding points use every two distinct members from the reference set. The three weighting schemes are also implemented with this algorithm. The data in Fig. 5 show that the version with four guiding points impairs the performance of both types of algorithms, and the A-type algorithms are more vulnerable to this impairment. We conjecture that this is because the guidance information becomes diluted and confounded when we consider too many distinct guiding points at the same time, causing the particle's search to acquire a degree of aimlessness. Similar phenomena have been observed in previous relevant researches, as previously noted.

V. CONCLUSIONS

In this paper we have proposed two primary types of novel learning strategies derived from principles embodied in the SS/PR template: (1) learning from every member of the swarm and (2) learning from every member of the reference set (where (1) may be viewed as a "soft version" of (2) that selects the reference set to consist of the entire swarm). The resulting Scatter PSO methods are shown to perform significantly better than the commonly used constriction coefficient PSO. Our algorithms select a small set of essential guiding points and consider multiple perspectives when influencing the individual particle's behavior. The number of guiding points should be in an appropriate range to make the guidance information clear and unambiguous; in our experiments the version with three guiding points works best on all functions tested. Between the approaches of (1) and (2), we establish that the exploitation of the reference set in (2) yields more robust outcomes under various experimental conditions.

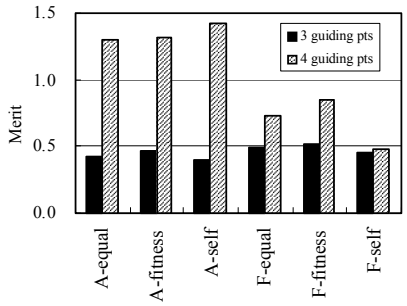

Fig. 5. Comparative performance between the algorithms with 3 and 4 guiding points.

Our findings suggest that the incorporation of additional strategic notions from scatter search and path relinking provide Scatter PSO hybrids that are still more effective, thus affording a promising avenue for future research.

REFERENCES

[1] J. Kennedy and R.C. Eberhart, Particle swarm optimization, *Proceedings IEEE Int'l. Conf. on Neural Networks*, *IV*, (1995) 1942-1948.

[2] R.C. Eberhart and Y. Shi, Evolving artificial neural networks, *Proceedings Int'l. Conf. on Neural Networks and Brain,* (1998) 5-13.

[3] V. Tandon, Closing the gap between CAD/CAM and optimized CNC end milling, Master thesis, Purdue School of Engineering and Technology, Indiana University Purdue University Indianapolis, (2000).

[4] H. Yoshida, K. Kawata, Y. Fukuyama, and Y. Nakanishi, A particle swarm optimization for reactive power and voltage control considering voltage stability, *Proceedings Int'l. Conf. on Intelligent System Application to Power Systems,* (1999) 117-121.

[5] N. Shigenori, G. Takamu, Y. Toshiku, and F. Yoshikazu, A hybrid particle swarm optimization for distribution state estimation, *IEEE Transaction on Power Systems*, 18, (2003) 60-68.

[6] P.Y. Yin, A discrete particle swarm algorithm for optimal polygonal approximation of digital curves, *Journal of Visual Communication and Image Representation,* 15, (2004) 241-260.

[7] R. Mendes, J. Kennedy, nd J. Neves, The fully informed particle swarm: simpler, maybe better, *IEEE Transaction on Evolutionary Computation*, 8, (2004) 204-210.

[8] F. Glover, A template for scatter search and path relinking, *Artificial Evolution*, Lecture Notes in Computer Science 1363, (1998) 13-54.

[9] F. Glover, M. Laguna and R. Marti, Fundamentals of scatter search and path relinking, *Control and Cybernetics*, 29 (3), (2000) 653-684.

[10] J. Kennedy and R. C. Eberhart, A discrete binary version of the particle swarm algorithm, *Proceedings IEEE Int'l. Conf. on Systems, Man, and Cybernetics,* (1997) 4104-4108.

[11] M. Clerc and J. Kennedy, The particle swarm explosion, stability, and convergence in a multidimensional complex space, *IEEE Transaction on Evolutionary Computation*, 6, (2002) 58-73.
[12] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, Handling multiple objectives with particle swarm optimization, *IEEE Transaction on Evolutionary Computation*, 8, (2004) 256-279.
[13] V. Campos, F. Glover, M. Laguna, and R. Martí, An experimental evaluation of a scatter search for the linear ordering problem, *Journal of Global Optimization*, 21 (4), (2001) 397-414.