Towards Swarm Diversity: Random Sampling in Variable Neighborhoods Procedure Using a Lévy Distribution

Gonzalo Nápoles[1], Isel Grau[2], Marilyn Bello[1], and Rafael Bello[1]

[1]Artificial Intelligence Laboratory, Universidad Central "Marta Abreu" de Las Villas, Cuba

[2]Bionformatics Laboratory, Universidad Central "Marta Abreu" de Las Villas, Cuba

{gnapoles, igrau, mbello, rbellop}@uclv.edu.cu

Abstract. Particle Swarm Optimization (PSO) is a non-direct search method for numerical optimization. The key advantages of this metaheuristic are principally associated to its simplicity, few parameters and high convergence rate. In the canonical PSO using a fully connected topology, a particle adjusts its position by using two attractors: the best record stored for the current agent, and the best point discovered for the entire swarm. It leads to a high convergence rate, but also progressively deteriorates the swarm diversity. As a result, the particle swarm frequently gets attracted by sub-optimal points. Once the particles have been attracted to a local optimum, they continue the search process within a small region of the solution space, thus reducing the algorithm exploration. To deal with this issue, this paper presents a variant of the Random Sampling in Variable Neighborhoods (RSVN) procedure using a Lévy distribution, which is able to notably improve the PSO search ability in multimodal problems.

Keywords. Swarm diversity, local optima, premature convergence, RSVN procedure, Lévy distribution.

Hacia la diversidad de la bandada: procedimiento RSVN usando una distribución de Lévy

Resumen. Particle Swarm Optimization (PSO) es un método de búsqueda no directo para la optimización numérica. Las principales ventajas de esta meta-heurística están relacionadas principalmente con su simplicidad, pocos parámetros y alta tasa de convergencia. En el PSO canónico usando una topología totalmente conectada, una partícula ajusta su posición usando dos atractores: el mejor registro almacenado por el individuo y el mejor punto descubierto por la bandada completa. Este esquema conduce a un alto factor de convergencia, pero también deteriora la diversidad de la población progresivamente. Como resultado la bandada de partículas frecuentemente es atraída por puntos sub-óptimos. Una vez que las partículas han sido atraídas hacia un óptimo local, ellas continúan el proceso de búsqueda dentro de una región muy pequeña del espacio de soluciones, reduciendo las capacidades de exploración del algoritmo. Para tratar esta situación este artículo presenta una variante del procedimiento Random Sampling in Variable Neighborhoods (RSVN) usando una distribución de Lévy. Este algoritmo es capaz de mejorar notablemente la capacidad de búsqueda de los algoritmos PSO en problemas multimodales de optimización.

Palabras clave. Diversidad de la bandada, óptimos locales, convergencia prematura, procedimiento RSVN, distribución de Lévy.

1 Introduction

The PSO metaheuristic [1, 2] is a well-known Swarm Intelligence method, which is able to solve challenging real-parameter problems from a distributed perspective, without a centralized control of a specific agent. Each organism in the swarm (hereinafter called particle) adjusts its position by using a combination of an attraction to the best solution that they individually have found, and an attraction to the best solution that any particle has found in its neighborhood [3], imitating those who have a better performance.

Following the above reasoning, the swarm overflies the solution space detecting promising regions. The features of this search method for dealing with global optimization problems include the improved ability of solving complex problems, high convergence speed and good generality for different problems [4].

Ironically the high convergence rate inherent to PSO algorithms using a fully connected topology (which means that a particle gets information of entire swarm) is the main cause of deteriorating the swarm diversity. As a result, the population is often attracted to stable points that are not necessarily global optima [5]. This behavior causes premature convergence of the algorithm, in which the particles are grouped about suboptimal solutions with little chance of escaping from this configuration. Once the particles have converged prematurely, they continue converging within extremely close proximity of one another so that the global best and all personal bests are within a very small region of the solution space [6], heavily limiting the exploration of new regions.

Several algorithms have been proposed in the literature to overcome this problem. For instance, niching or multimodal methods have been developed to reduce the undesirable effects of genetic drift [7]. They are able to compute not just one, but many local or global solutions in a single run, preserving the population diversity and also allowing parallel convergence. Inspired on niching, Chen and Montgomery [8] proposed a simple strategy to reduce crowding in PSO-based algorithms, which leads to significant performance improvements in multimodal search spaces. Unfortunately, this strategy is ineffective for solving unimodal problems. In Section 3 we provide a more detailed review of existing approaches for handling premature convergence, resulting from the progressive deterioration of the diversity.

Another research direction is oriented to particle interaction. Perhaps the most important variation of the standard PSO is the inception of the *lbest* swarm model or local topology [9]. Here a particle gets information of only a subset of the whole population. Consequently, multiple swarms may coexist and thus many global attractors are used to guide the search. It is the main difference with respect to the *gbest* or fully connected topology where particles obtain information from any agent in the artificial population. The "advantage" of the *lbest* model in comparison to the *gbest* model is its slow convergence rate, which helps to avoid premature convergence states. Nevertheless, it is the slower rate of convergence of the *lbest* model that is most responsible for the general disregard of it as an alternative up to this point [3].

In summary, many attempts to increase the diversity in the artificial population perform well for multimodal search spaces, but they are unable to efficiently solve unimodal problems. On the other hand, the much faster convergence of the *gbest* model seems to indicate that it produces superior performance, but this may be misleading [3] for solution search spaces having large number of optima. Obviously it is desirable to formulate new approaches capable to solve both unimodal and multimodal problems, but preserving the high convergence speed distinctive in PSO algorithms using a fully connected topology.

Recently the authors of this work introduced a relatively simple method called Random Sampling in Variable Neighborhoods (RSVN) for controlling the swarm diversity [10, 11]. The main idea of this procedure is to restructure the population from the selection of random samples uniformly distributed in several neighborhoods, whenever a stagnation or premature convergence is detected. This paper introduces two main improvements in the RSVN algorithm: (i) we replace the uniform distribution for a Lévy distribution allowing better exploration in the sampling task, and (ii) we incorporate two new operators for recombination and selection. Finally, we obtain a suitable mechanism for controlling the swarm diversity, outperforming other algorithms reported in the literature, for both unimodal and multimodal problems.

The rest of the paper is organized as follows: in the next section a theoretical background of standard PSO is discussed. In Section 3 we provide a review of PSO algorithms designed to handle non-progress states. In Section 4 the RSVN procedure is described. Section 5 gives details about the proposal, whereas Section 6 reports simulations and statistical analysis for

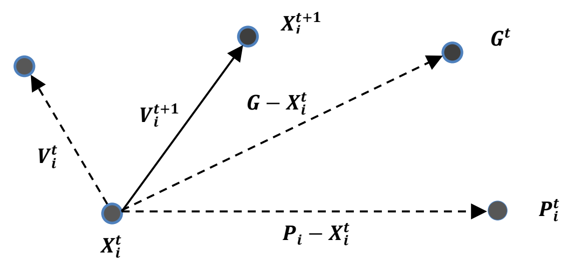

Fig. 1. PSO rules for particle trajectory in each step

benchmark problems. Finally, conclusions and research aspects are presented in Section 7.

2 Particle Swarm Optimization

The PSO metaheuristic is a robust non-direct global-search method for solving challenging continuous optimization problems. It involves a set of agents (particles) known as swarm which explore the space trying to locate promising regions [1, 2]. Swarm particles are interpreted as solutions for the optimization problem and they are codified as n-dimensional vectors. In the case of canonical PSO, each particle X_i explores the space using its own velocity V_i, a local memory of the best position it has obtained P_i and knowledge of the best solution G found in its neighborhood. For a graphical illustration of this idea, Fig. 1 shows the rules for updating the particle trajectory at each step t of the search process.

In the standard algorithm, particle's velocities require to be clamped at a maximum value $Vmax$; without this clamping the swarm is prone to entering in a state of *explosion* (i.e., the state where the velocity components and positional coordinates careen toward infinity) [12]. However, estimating this parameter is a complex task since very large spaces require larger values to ensure a correct exploration, while smaller search spaces require small values avoiding the swarm explosion.

In addition, PSO requires specification of two positive factors c_1 and c_2 denoting the *cognitive* and the *social* factors, respectively. The acceleration coefficients determine the magnitude of the forces in the direction of the personal best record P_i and the neighborhood best solution G_i. It also should be mentioned that the behavior of the PSO metaheuristic changes radically with these coefficients values [13], which could cause stagnation or cyclic behavior in particle's trajectory.

Equations 1 and 2 show how to update particle's position and velocities based on the above components. Here k indexes the current cycle, $\phi_1 \sim U(0, c_1)$ and $\phi_2 \sim U(0, c_2)$ are two random numbers with uniform distribution, whereas ω denotes the *inertia weight* designed by Shi and Eberhart [14] to replace the parameter $Vmax$ by adjusting the influence of the previous particle velocities on the optimization process.

The introduction of this inertia weight guarantees an apposite balance between local and global search. Higher weights will facilitate the exploration of new regions of the solutions space, while lower weights will facilitate the local search. But the incorrect choice of this factor could negatively affect the algorithm performance, so it is recommended to adjust it dynamically during the progress of the search procedures [11].

$$V_i^{t+1} = \omega V_i^t + \phi_1 (P_i - X_i^t) + \phi_2 (G - X_i^t) \tag{1}$$

$$X_i^{t+1} = \mathrm{X}_i^t + V_i^{t+1} \tag{2}$$

Another feature introduced to guarantee, in a certain sense, a proper balance between exploration and exploitation is known as *constriction*. It was designed by Clerc and Kennedy [12] for preventing explosion, ensuring the algorithm convergence and eliminating $Vmax$. In the Clerc's work different ways to implement the constriction coefficient (χ) are discussed. The constriction termed Type 1 is perhaps the simplest method; it is derived from factors c_1 and c_2 as Equation 3 suggests.

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \varphi = c_1 + c_2 \tag{3}$$

It was observed that when $\varphi < 4$, then the swarm would slowly "spiral" toward and around the global-best solution found during the execution with no guarantee of convergence, whereas for $\varphi > 4$ convergence would be quick and guaranteed. Commonly the acceleration coefficients c_1 and c_2 are set to 2.05 implying that

$\varphi = 4.1$, so the constant χ is approximately 0.729822. A comparison study of both methods demonstrated that the constriction is in fact a special case of the algorithm with inertia weight in which the parameters have been determined analytically [10].

For better understanding of this asseveration, let's consider Equation 4 where the rule for updating the velocities of particles in the swarm is modified, by the inception of the constriction coefficient as suggests the Type 1 implementation discussed above. Notice that the previous velocity as well as the differentials $\phi_1(P_i - X_i)$ and $\phi_2(G - X_i)$ are multiplied by $\chi = 0.729822$. Assuming that the coefficients c_1 and c_2 are fixed, we can conclude that $\omega = 0.7298$ and $c_1 = c_2 \approx 1.4961$. Taking this into account, the constricted swarm will converge without specifying any $Vmax$ at all.

$$V_i^{t+1} = \chi\left(V_i^t + \phi_1(P_i - X_i^t) + \phi_2(G - X_i^t)\right) \tag{4}$$

Conversely, experiments reported in [14] concluded that a more prudent approach is to clamp velocities components of particles anyway. For example, $Vmax$ could be taken as the half of the range of each problem variable. As a result, we obtain a PSO algorithm with no-problems specific parameters (only the particles number and generations are required). Actually, this canonical optimizer using the inertia weight (analytically determined as it was explained) will be adopted in all the simulations described in next sections.

2.1 Swarm Organization

A key aspect that should be discussed is how to define the particle's neighborhoods in order to select the global-best particle at each cycle. Here, topological neighborhoods of a swarm particle are referring to the protocol in the particle's communication. The most popular approaches are grouped in two major groups: the *gbest* or global topology, and the *lbest* models or local topology.

In the first one, agents use global knowledge obtained from the whole swarm; which means that at each step the best solution G is updated as the particle having the best fitness value, from the entire population. From a geometric point of view, this model may be conceptualized as a fully connected graph. In general, the *gbest* PSO is more accurate for solving unimodal problems [3], showing a high convergence rate. But it often shows a poor performance in multimodal problems, since the swarm tends to prematurely converge towards sub-optimal solutions in a few iterations.

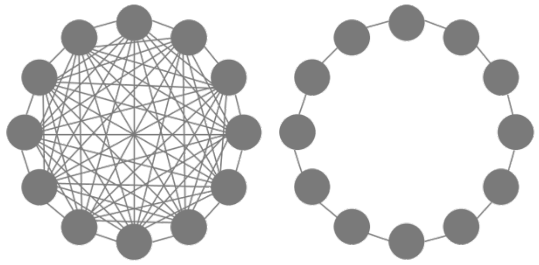
Fig. 2. Particle's interaction: a) in the *gbest* model the particles are fully connected and they share information among all individuals; b) in the *lbest* ring topology each particle only interacts with its immediate neighbors

In the case of *lbest* topology, a particle shares information with a limited subset of the swarm. It induces a partition of the global population in multiple sub-populations, where each of them has its own topological global-best record. More precisely, the *lbest* model involves not just a single global best registry, but several topological global-best solutions G_i are presented. There exist many implementations of the *lbest* model, but perhaps the most widely used is the *lbest ring topology*, where each particle is able to interact with its immediate neighbors (using the population indices to identify the left and right neighbor of each particle). Fig. 2 illustrates the interaction among particles in a *gbest* model, and also in an *lbest* model using its simplest implementation: a ring topology.

The *lbest* topology has the advantage of allowing parallel search, thus subpopulations could converge to diverse regions of the solution space. This desirable feature resulted in a more thorough search strategy [13]. Overall *lbest* converges more slowly than the *gbest* topology, being less vulnerable to the attraction of local optima. In [9] several communication structures such as circles, wheels, starts, etc. were extensively studied.

From this research, the relative superiority of the von Neumann structure was concluded. It has the parallelism of *lbest*, but uses 4 nodes for the individual's communication. Hence, the graph is more densely connected that the *lbest* strategy using a ring topology, but still less connected than *gbest*. Despite this result, it is known that the correct selection of a specific topology is conditioned by the problem properties, but in most cases this knowledge is unknown in advance.

3 Handling Stagnation and Premature Convergence States: a Brief Review

In this section we provide a concise review of some PSO-based algorithms which were specially designed for preserving the swarm diversity during the optimization progress. These models allow handling potential stagnation and premature convergence situations, thus offering in many cases a proper escaping alternative from local optima.

3.1 Bare Bones Particle Swarm

Bare Bones Particle Swarm (BBPSO) was designed by Kennedy as a PSO-based model for dynamic problems. Here, dependence on velocity is replaced by sampling from a Gaussian distribution. This approach is supported by the fact that particles converge to a weighted average between their personal best and neighborhood best positions [15]. Hence, in order to understand the particles' behavior and also to identify the similarity features with respect to other stochastic population-based optimizers, Kennedy [16] proposed a modified algorithm without using the velocity formula in the update equation as follow:

$$x_i \sim N\left(\frac{G + P_i}{2}, |P_i - G|\right) \tag{5}$$

where G is the best informer in the neighborhood of the i-th particle and P_i is the personal best peak discover by the individual. Gaussian bare-bones works quite well, imitating the performance of PSO on some problems, but proving less effective on other [17]. On closer examination, what appears to be a bell curve actually has a kurtosis which increases with iteration, and the distribution has fatter tails than the Gaussian generator.

It has been suggested that the origin of this lies in the production of "bursts of outliers" [18]. In fact, Kennedy discovered that if burst events are added to Gaussian bare-bones, the performance is notably improved. The conjecture, thus, is that the fat tails in the position distribution of canonical PSO enhance the ability of the swarm to move from sub-optimal locations [13]. From this observation it is clear that the Gaussian generator may be replaced by a more explorative distribution.

3.2 Comprehensive Learning PSO

Comprehensive Learning PSO (CLPSO) [19] was specifically designed for optimizing complex multimodal problems. In this implementation, the social component from the velocity equation is removed. Also, the velocity of each particle component has the chance to be influenced by the best record of every other particle, thus encouraging the diversity of the swarm. The expression for updating the velocity is summarized in Equation 6, where $fi(d)$ defines the personal-best position that will be used for updating the d-th dimension of the velocity of the i-th particle.

$$V_i^{t+1} = \omega V_i^t + \phi_1\left(P_{fi(d)} - X_i^t\right) \tag{6}$$

A learning probability $LP_i \in [0.05, 0.5]$ decides whether a velocity will be updated using its own personal-best or not. More explicitly, a set of particles for updating the velocity component of each particle (called exemplars) is selected by using a uniform random number in $(0,1)$ for each dimension of the current particle. If this random number is greater than LP_i then the corresponding velocity dimension will be updated using its own personal-best position computed during the search, otherwise other personal-best record will be used.

In the latter version, two particles are randomly chosen from the swarm (excluding the current one), and then the one with better personal value is used for updating the current dimension. This process is iteratively repeated for each dimension of the i-th particle. In summary, CLPSO performs well for multimodal problems, but it has poor performance against other algorithms on unimodal or relatively simple multimodal functions.

3.3 Opposition Based PSO

The main idea of OPSO [20] is to use an opposition based learning method and a dynamic Cauchy mutation operator to help avoid local optima and accelerate the convergence of the PSO algorithm. Opposition based learning method was introduced by Tizhoosh [21] and it has proven to be a convenient principle for solving global optimization problems. When evaluating a solution x to a given problem, we can suppose the opposite solution of x to get a better solution x'. By doing this, the distance of x from optima solution can be reduced. Besides, boundaries should be updated dynamically for controlling the step size of opposition. The minimum (a_j^*) and maximum (b_j^*) values of each dimension in current population are used for computing the opposite solution instead of a predefined interval $[a, b]$. Equation 7 shows how to compute the opposition-based method.

$$Ox_i^j = a_j^* + b_j^* - x_i^j \tag{7}$$

In OPSO the opposite swarm particle is calculated according to a probability p_o, then n fittest particles from the current swarm and the opposite swarm are selected as survivors. Also, at each cycle the best particle is mutated using a dynamic Cauchy mutation operator [20], trying to update the global best particle in the swarm. Simulations shown that OPSO has faster convergence on simple unimodal functions, and better global search capability on multimodal functions compared to the standard PSO. However, there are still fewer cases where OPSO falls in the local optima.

3.4 Attraction-Repulsion PSO

As an attempt of controlling the swarm diversity from particles movements and trying to overcome the problem of premature convergence, in [22] the authors propose the Attractive and Repulsive PSO (ARPSO). This PSO-based algorithm alternates between phases of attraction and repulsion. As a result, it defines the attraction phase merely as the basic PSO algorithm, where particles will attract each other due to the information flow of good solutions among particles, while in the repulsion phase the velocity-update formula of particles is inverted as follows:

$$V_i^{t+1} = \omega V_i^{\mathrm{t}} - \phi_1(P_i - X_i^{\mathrm{t}}) - \phi_2(G - X_i^{\mathrm{t}}) \tag{8}$$

In the repulsion phase an individual particle is no longer attracted to, but instead repelled by the best known particle position G and its own previous best position P_i; whereas in the attraction phase the swarm is contracting, and consequently the diversity decreases. More explicitly, when the diversity drops below a lower threshold d_low the algorithm switches to the repulsion phase, where the swarm expands due to the above inverted update-velocity Equation 9. On the other hand, when a diversity of d_high is reached, the ARPSO switches back to the attraction phase. The diversity measure used in ARPSO is defined as

$$div(S) = \frac{1}{|S| * |L|} \sum_{i=1}^{|S|} \sqrt{\sum_{j=1}^{N} \left(p_{ij} - \bar{p_J}\right)^2} \tag{9}$$

where S denotes the swarm, $|S|$ is the swarm size, $|L|$ is the length of the longest diagonal in the search space, N is the dimensionality of the problem, p_{ij} is the j-th value of the i-th particle, while $\bar{p_J}$ is the *j*-th value of the average point p. It is remarkable that this diversity measure is independent of the swarm size, the dimensionality as well as the search range in each dimension.

3.5 Quadratic Interpolation Based PSO

The QPSO method [23] introduces a quadratic crossover operator in the basic algorithm, with the hope to increase the diversity. This operator is a nonlinear multi-parent crossover operator which makes use of three swarm particles (parents) to produce a new agent (offspring) which lies at the minimum of the quadratic curve passing through the three selected individuals. The j-th dimension of the new individual is determined as Equation 10 suggests, where a denotes the particle having minimum evaluation, while b and c are two randomly selected particles from the swarm.

$$\tilde{x}_i^j = \frac{\left(b^{j^2} - c^{j^2}\right)f(a) + \left(c^{j^2} - a^{j^2}\right)f(b) + \left(a^{j^2} - b^{j^2}\right)f(c)}{2[(b^j - c^j)f(a) + (c^j - a^j)f(b) + (a^j - b^j)f(c)]} \tag{10}$$

At that point, the offspring particle $\tilde{x}$ replaces the worst individual in the swarm, regardless of its quality. In this way the search is not limited to a region around the current best location, but is in fact more diversified in nature. Empirical results show that, for a reduced benchmark suite, QPSO is able to preserve the swarm diversity. Also, the authors observed that QPSO is quite competent for solving problems of dimensions up to 50, which leads to a highly desirable property.

4 Controlling Swarm Diversity through a Novel Reorganization Procedure

As it was discussed above, many modifications to the *gbest* model have been introduced with the aim to preserve the population diversity, providing an alternative to escape from local optima. It thus prevents such states where the objective function does not suffer improvements. In most cases the *gbest* model is the selected optimizer because it shows high convergence rate against other algorithms. Although such approaches frequently are able to outperform the original PSO, they report poor performance for a considerable subset of benchmark functions [11]. Hence, there is a need to incorporate more competent models in order to enhance the PSO search process.

Having this idea in mind, in [10, 11] the authors introduced a procedure for controlling the swarm diversity. It is called Random Sampling in Variable Neighborhoods (RSVN). The key idea of this procedure is to restructure the swarm from the selection of random samples uniformly distributed in M neighborhoods, generated around the best solution found so far. More explicitly, the proposed RSVN algorithm has two steps: the first one is related to the detection of premature convergence or stagnation states, while the second is oriented to the swarm reorganization. In next subsections we explain this method in detail.

4.1 Detection of Non-Progress States

One of the causes of premature convergence in the basic PSO is the poor swarm diversity [22]. When PSO falls into a local optimum all individuals are grouped around this solution, that is why diversity is gradually lost among particles, making it more difficult to find better solutions. Diversity can be used to monitor the swarm behavior, i.e., the degree of convergence or divergence [24].

The first step of the RSVN procedure consists of detecting potential non-progress states, that is, stagnation or premature convergence situations. In these states the swarm is unable to find better solutions, and thus the optimization of the evaluation function doesn't report significant progress in consecutive cycles. However, how to efficiently detect these non-progress states?

Several diversity measures have been developed for detecting possible states of premature convergence. For example, in [25] the authors studied several diversity measures including a) the swarm radius, b) the average distance around the swarm center, c) the normalized average distance around the swarm center, d) the average of the average distance around all particles in the population, and e) swarm coherence (defined as the swarm center divided by the average speed of all particles in the swarm).

In the PSO-RSVN-α algorithm, the radius of the swarm is adopted as a diversity measure. Equation 11 mathematically formalizes the

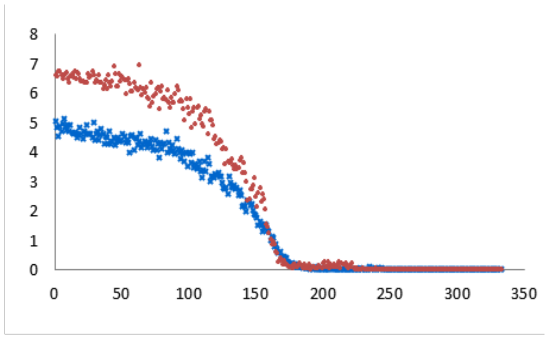

Fig. 3. Swarm diversity measures applied to the well-known Shifted Rosenbrock's problem

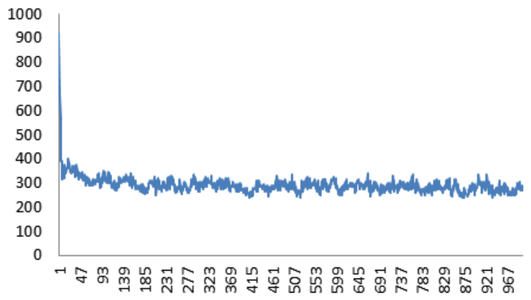

Fig. 4. Simulation of the average distance around the swarm center for the problem F_3

normalized expression of this measure, where $||X||$ denotes the Euclidean norm, $Xmin$ and $Xmax$ are the end points on which each dimension of the particle X_i is defined (for simplicity in the notation we assume the same domain for all dimensions), whereas S denotes the swarm. Hence, the closer to zero the swarm radius, the more crowded the particles, implying poor exploration ability.

$$\rho(t) = \frac{\max_{1 \leq i \leq |S|} \left\| X_i^{(t)} - G \right\|}{|Xmax - Xmin|} \tag{11}$$

Fig. 3 shows the behavior of two diversity measures for Shifted Rosenbrock's function (see benchmark description in [26]). More explicitly we explore the *swarm radius* (upper group of points) and also the *average distance around the swarm center* (lower group of points). In this example, a *gbest* model with inertia weight and standard parameters is used. Observe how the swarm diversity is gradually lost throughout the evolution of the search process.

Then the PSO-RSVN-α algorithm restructures the population if the diversity measure is less than a fixed threshold α. However, this approach may be ineffective for problems where particles are not able to easily discover promising regions. In other words, in stagnation states the particle swarm is not necessarily crowded in a small region of the search space, which is the main difference with a population that is prematurely attracted to a local optimum. In such cases the swarm is not able to detect a local solution and thus the objective function doesn't have significant improvement.

For instance, let's analyze the *Shifted Rotated High Conditioned Elliptic Function* [26]. It is a very difficult unimodal problem where the optimum was shifted and also the problem was rotated. Fig. 4 illustrates the behavior of the average distance around the swarm center; notice that the swarm diversity is lost in a few iterations, but subsequently this degradation remains "stable" during the rest of the search. But in our simulations the evaluation of the global-best particle doesn't show any improvement, which means that probably the population suffers from a stagnation state.

In order to detect potential stagnation configurations, in [11] the PSO-RSVN-P implementation is introduced. This method uses a simple and low time-consuming criterion: the particle swarm is in a stagnation state if the objective function remains without improvements in P iterations. Notice that it could be also effectively used to detect premature convergence states. In this sense a central drawback should be solved: how to select an appropriate approach for automatically detecting a specific non-progress state without previous knowledge of the objective function landscape?

In this paper we use a hybrid strategy where a non-progress state (whatever it is) is declared if (1) the average distance around the swarm center is less than a threshold α, or (2) the objective function remains without improvements in P cycles. It should be additionally remarked that in the present work the swarm radius is replaced by the average distance around the swarm center, since in [25] Olorunda and Engelbrecht demonstrated that this measure is less sensible to the effects of outliers, thus it is more robust. Precisely the first proposal attempting to improve the PSO-RSVN algorithm is related to this simple modification.

4.2 Particle Swarm Reorganization

Once premature convergence signals are detected it is necessary to take some action to allow the algorithm to escape from this state. Here several approaches may be adopted. For example, in [6] Evers and Ghalia utilize a scheme based on Simulated Annealing [27] for moving the global-best attractor in the swarm, i.e., the position of the global-best individual found so far. Although it reported large improvements for a relatively simple set of problems, in some problems it is difficult to locate a better particle using local search, and even when once found, the swarm is poorly diversified. For this reason it is clear that another research direction to deal with the issue is required.

Reorganizing the population is also a feasible alternative. Diversification increases the chances to escape from the local optimum but if the swarm is not properly reorganized, it may converge toward the same solution or indiscriminately move away from promising areas that were detected [11] which instead of benefiting could affect the search performance. More recently, we developed a simple but effective procedure called Random Sampling in Variable Neighborhoods (RSVN). It tries to restructure the swarm when a premature convergence or stagnation state is detected.

This algorithm has two central steps: (1) the generation of some samples uniformly distributed around the global-best individual found so far, and (2) the elitist selection of the survivors that should be present in the following population. It induces further diversity over the swarm, increasing the chance to find better solutions. The operators of the RSVN algorithm are described as follow.

Equation 12 shows how to generate a sample x_i belonging to the k-th neighborhood, where G is the global-best individual found so far at current iteration, while R denotes the particle's range, that is, $R = Xmax - Xmin$. For simplicity we consider the same domain for all dimensions of the problem. In addition, this equation involves a parameter $\xi_k \in (0,1]$ representing the k-th neighborhood proportion with respect to the size of the search space. It is calculated as $\xi_k = k/M$, where M is the number of partitions. Following this logic, in each neighborhood a set of samples Ψ_k is generated. It will be remarked that closest neighborhoods provide local search or exploitation, while largest ones ensure better exploration.

$$x_i^k \sim U(G - \xi_k R, G + \xi_k R) \tag{12}$$

Once the set of samples has been generated at each neighborhood, a strong elitist selection process takes place in order to efficiently restructure the particle swarm. It means that for each set of samples Ψ_k a good enough subset Φ_k is then selected as a survivor of the next population. Here good enough particles refer to individuals having better evaluation of the objective function, but a different strategy may be adopted. Thus, selected agents will comprise the reorganized swarm. This scheme is summarized as follows:

$$S = \Phi_1 \cup \Phi_2 \cup \ldots \cup \Phi_M = \bigcup_{k=1}^{M} \Phi_k \quad : \Phi_k \subseteq \Psi_k, \forall k \,. \tag{13}$$

Obviously the global-best particle found at the current iteration should be also preserved as survivor in the next population, thus it must replace the worst particle in the reorganized swarm. Due to its simplicity, elitist properties and relatively low computational cost, the RSVN procedure summarized here could be easily adapted and integrated into other population based search methods.

Actually, the PSO-RSVN-P variation has been successfully used for learning the weight matrix of a Fuzzy Cognitive Map characterizing the biological behavior of HIV proteins [28]. This scheme is also capable to predict the resistance of *protease* mutations to existing drugs, largely outperforming conventional classifiers such as

Neural Networks, Support Vector Machine or Decision Trees.

5 Improving the RSVN Procedure

Despite the success of the RSVN method for handling non-progress states, we consider that this procedure could be enhanced. For example, is the uniform distribution in the sampling process the best choice? For dealing with such issues, this section introduces two modifications to the RSVN algorithms. First, the uniform distribution used in the random sampling process is replaced by a Lévy distribution; it allows exploring the search space more efficiently. In addition, we introduce a new operator based on Differential Evolution [29] to recombine the sets of samples, in order to guarantee better diversification.

Several studies reported that the flight behavior of some animals and insects in nature is related to the Lévy distribution [30, 31]. For example, some species of eagles such as Golden Eagles or *Aquila Chrysaetos* explore their territory by flying in a random manner much like the Lévy flights [31] in order to locate the prey. In general, algorithms employing Lévy distribution have a long fat tail; then they are more capable of escaping from local solutions [17, 32]. That's why this distribution seems to be a proper alternative for replacing the uniform generator in the sampling process.

The Lévy probability distribution is a class of probability distribution having an infinite second moment and governing the sum of these random variables [33, 34]. Formula 14 shows the form of this distribution. It is symmetric with respect to $y = 0$ and has two parameters γ and α, where the first factor represents the scaling factor satisfying $\gamma > 0$, whereas $0 < \alpha < 2$ is used for controlling the shape of the distribution.

$$L_{\alpha,\gamma} = \frac{1}{\pi}\int_0^\infty e^{-\gamma q^\alpha}\cos(qy)dq \quad , y \in \mathbb{R} \tag{14}$$

The analytic form of the integral is unknown for general α, but is well-known for a few cases (for $\alpha = 2$ it is equivalent to Gaussian distribution, whereas for $\alpha = 1$ it is equivalent to Cauchy distribution). Fortunately in [35] the authors present a numerical algorithm for generating Lévy random numbers. In present work, we make use of this method to compute numbers with Lévy distribution, where the value for α is set to 0.8.

Equation 15 summarizes the samples generation rule using the new probability distribution. Observe that the search ranges were also modified since in Equation 12 we can't guarantee that generated samples are feasible (they are not necessarily within the domain range) and hence particles require being restricted. It could cause a situation where many individuals are enclosed in a small region around the boundary. To overcome this disadvantage we define $R_1 = G - Xmin$ and $R_2 = Xmax - G$ which definitely ensure generating feasible samples.

$$x_i^k \sim L(G - \xi_k R_1, G + \xi_k R_2) \tag{15}$$

As a second modification of the RSVN algorithm, this work introduces a recombination operator based on Differential Evolution rules. Let Ψ be the set of all swarm particles that were sampled in neighborhoods of the global-best point at the first stage of the original RSVN procedure. Also suppose that $|\Psi| = |S|$, then the set Φ of mutants is now computed by reproducing collected samples as Formula 16 suggests. This operator is inspired on the well-known differential strategy DE/current-to-rand/1 without crossover [29], which introduces a perturbation on the search mainly based on the knowledge of collected samples, complementing the swarm reorganization mechanism.

$$x_i' = G + F(a_1 - a_2), \;\; a_1, a_2 \in \Psi \tag{16}$$

In the above equation, a_1 and a_2 are two randomly selected agents taken from the set of samples Ψ, whereas the factor $0 < F \leq 1$ is used for controlling the differential amplification. Small values for this factor (e.g. $F < 0.2$) become better exploitation, but large values (e.g. $F > 0.8$) ensure higher exploration. For simplicity, this work assumes that the value for the parameter F is fixed to 0.5. As a result of this process, a set Φ of mutants having the same cardinality of Ψ is obtained, that is, the number of particles in the swarm.

Table 1. Average error computed for unimodal, basic multimodal and expanded problems (F1-F14) over 30 runs

ID	GBest	LBest	BBPSO	arPSO	QPSO	OPSO	CLPSO	RSVN	LSVN
F_1	9.701E-1	6.331E-1	3.79E-15	1.031E-1	2.396E+0	2.748E+0	3.611E-3	8.670E-7	1.973E-9
F_2	9.770E+2	3.793E+2	1.12E-13	6.362E+2	7.407E+2	3.685E+2	2.202E+1	5.760E-4	6.456E-7
F_3	2.031E+6	1.248E+6	1.666E+5	8.366E+5	1.570E+6	1.377E+6	2.015E+6	3.213E+5	1.114E+5
F_4	7.867E+3	4.690E+3	1.140E-5	7.719E+3	6.846E+3	5.007E+3	3.265E+1	2.163E-1	3.232E-2
F_5	2.070E+3	3.395E+3	2.50E-11	2.152E+3	2.381E+3	1.975E+3	3.147E+2	1.475E+3	3.837E+2
F_6	2.514E+3	4.347E+3	1.142E+0	1.973E+3	2.784E+3	1.483E+3	4.739E+2	9.790E+0	4.299E-1
F_7	1.330E+3	1.398E+3	1.263E+3	1.330E+3	1.346E+3	1.345E+3	1.286E+3	1.298E+3	4.976E+0
F_8	2.017E+1	2.054E+1	2.036E+1	2.177E+1	2.016E+1	2.042E+1	2.037E+1	2.036E+1	2.039E+1
F_9	39.92E+1	2.916E+1	6.533E+0	4.046E+1	3.712E+1	3.912E+1	3.317E+1	3.980E+0	2.004E-1
F_{10}	7.050E+1	4.972E+1	2.368E+1	6.891E+1	7.030E+1	6.366E+1	3.906E+1	2.666E+1	1.767E+1
F_{11}	7.450E+0	6.471E+0	5.006E+0	7.035E+0	7.020E+0	7.028E+0	8.306E+0	5.285E+0	5.109E+0
F_{12}	9.613E+3	3.539E+2	5.225E+2	1.279E+4	1.533E+4	1.145E+4	3.148E+3	6.442E+3	4.507E+2
F_{13}	1.751E+0	1.246E+0	6.607E-1	1.937E+0	1.560E+0	1.550E+0	3.790E+0	7.421E-1	4.892E-1
F_{14}	3.719E+0	3.265E+0	3.148E+0	3.538E+0	3.442E+0	3.276E+0	3.577E+0	2.952E+0	2.616E+0

Table 2. Average error computed for the composed multimodal functions (F15-F25) over 30 runs

ID	GBest	LBest	BBPSO	arPSO	QPSO	OPSO	CLPSO	RSVN	LSVN
F_{15}	4.210E+2	3.838E+2	3.104E+2	4.397E+2	4.726E+2	4.240E+2	5.556E+2	2.301E+2	2.350E+2
F_{16}	1.970E+2	1.564E+2	1.428E+2	1.722E+2	1.741E+2	2.029E+2	1.869E+2	2.663E+2	1.286E+2
F_{17}	1.800E+2	2.142E+2	1.457E+2	2.014E+2	2.026E+2	1.888E+2	2.008E+2	1.355E+2	1.422E+2
F_{18}	7.210E+2	3.897E+2	7.128E+2	6.875E+2	8.945E+2	7.831E+2	3.532E+2	7.518E+2	6.664E+2
F_{19}	6.470E+2	4.203E+2	8.662E+2	7.715E+2	9.592E+2	7.571E+2	3.551E+2	5.248E+2	7.176E+2
F_{20}	8.345E+2	5.331E+2	6.634E+2	6.557E+2	7.034E+2	7.827E+2	3.546E+2	5.934E+2	6.149E+2
F_{21}	1.201E+3	1.072E+3	8.933E+2	1.106E+3	1.043E+3	1.001E+3	5.227E+2	7.765E+2	4.955E+2
F_{22}	8.446E+2	7.726E+2	7.882E+2	8.012E+2	8.398E+2	7.033E+2	8.156E+2	7.123E+2	7.891E+2
F_{23}	9.629E+2	8.257E+2	1.100E+3	9.935E+2	1.060E+3	9.983E+2	6.070E+2	8.951E+2	7.982E+2
F_{24}	7.466E+2	2.681E+2	4.069E+2	5.399E+2	6.750E+2	9.066E+2	2.301E+2	3.767E+2	2.000E+2
F_{25}	1.810E+3	1.822E+3	1.735E+3	1.811E+3	1.809E+3	1.681E+3	1.774E+3	1.623E+3	4.255E+2

At the end, the selection of survivors takes place, which means that the updated swarm will be composed by $|S|-1$ agents taken from the set $S' = \Psi \cup \Phi$, having the best fitness value. Obviously the global-best particle should be included in the next swarm, with the goal of preserving the search knowledge obtained during the optimization steps.

6 Simulations and Statistical Analysis

In this section we compare the performance of the proposed algorithm (PSO-LSVN[1]) with respect to eight well-known variants: a *gbest* PSO with inertia weight, an *lbest* PSO using a Ring Topology and inertia weight, Bare Bones PSO, Attraction-Repulsion PSO, Quadratic Interpolation PSO, Opposition based PSO, and also we include the original PSO-RSVN. For all the cases the parameters' settings discussed in the original references are assumed. As well, all the PSO variants use an inertia weight fixed to $\omega = 0.7298$, where the coefficients c_1 and c_2 are fixed to 1.496.

In case of the PSO-RSVN and PSO-LSVN variants, we set the diversity threshold $\alpha = 1.0\mathrm{E}-5$, the allowed number of objective evaluations without progress $P = 100$, whereas for the sampling method ten variable neighborhoods ($M = 10$) are taken. In each simulation the number of particles in the swarm is 50. Likewise, all the algorithms are stopped when 100.000 evaluations of the evaluation function are reached or when the computed error is less than 1.0E-20.

As a benchmark suite, we have selected the 25 test problems of dimension 10 reported at the CEC'2005 special session on real parameter optimization [26]. This collection is composed by 5 unimodal functions (F_1-F_5) and 20 multimodal problems including 7 basic functions (F_6-F_{12}), 2 expanded problems (F_{13}-F_{14}) and also 11 hybrid functions where each one (F_{15} to F_{25}) has been defined through compositions of F_{10} out of the F_{14} previous functions. All functions were shifted to guarantee that their optima can never be located in the center of the search space.

Tables 1 and 2 summarize the averaged error computed over 30 trails for each algorithm, with respect to the benchmark suite. For better understanding these simulations are conducted as follows: Table 1 refers to unimodal, basic multimodal and expanded problems, whereas in Table 2 we detail the behavior of all the studied algorithms for the composed multimodal functions.

[1] Here PSO-LSVN refers to a *gbest* model using the improved RSVN procedure introduced before, leading to the main contribution of this work.

As a brief characterization of these results we can perceive that BBPSO and PSO-LSVN reported lowest errors (7050.20859 and 4702.9709) in comparison to other approaches. But we cannot conclude anything yet. In a deeper study of these algorithms, several statistical tests to explore significant differences among them are used. Depending on the concrete type of data employed, statistical tests can be arranged into two classes: parametric and non-parametric [36]. Unfortunately, parametric tests are based on assumptions (independence, normally, homoscedasticity) which are most probably violated when analyzing the global performance of stochastic algorithms based on computational intelligence [37, 38].

Then we need to apply non-parametric tests. As a first step, the Friedman procedure (Friedman two-way analysis of variances by ranks) [39] is computed, for detecting significant differences between two or more algorithms. In other words, it is a multiple comparisons test detecting whether at least two of the samples represent populations with different median values or not, in a set of n samples ($n \geq 2$). Table 3 illustrates the mean rank computed for the test. Using a confidence interval

Table 4. Results achieved by the Wilcoxon test

Algorithm1	Algorithm2	p-value[a]	Hipotesis
PSO-LSVN	BBPSO	0.002	Rejected
	PSO-RSVN	0.011	Rejected
	CLPSO	0.045	Rejected
	LBest	0.009	Rejected
	OPSO	0.000	Rejected
	arPSO	0.000	Rejected
	GBest	0.000	Rejected
	QPSO	0.000	Rejected
PSO-RSVN	BBPSO	0.904	Accepted
	CLPSO	0.840	Accepted
	LBest	0.115	Accepted
	OPSO	0.000	Rejected
	arPSO	0.000	Rejected
	GBest	0.000	Rejected
	QPSO	0.000	Rejected

[a] Using the Monte Carlo signification.

Table 3. Mean Rank achieved by the Friedman test

Algorithm	Mean Rank [a]	Std deviation
PSO-LSVN	2.16	2.224E+4
BBPSO	3.28	3.325E+4
PSO-RSVN	3.36	6.415E+4
CLPSO	4.60	4.030E+5
LBest	5.08	2.495E+5
OPSO	6.12	2.752E+5
arPSO	6.36	1.670E+5
GBest	7.00	4.059E+5
QPSO	7.04	3.138E+5

[a] Monte Carlo signification (p-value) = 0.000

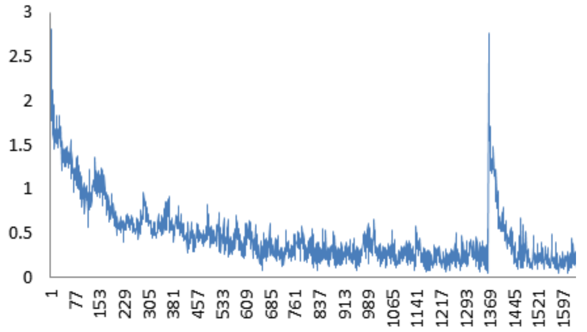

Fig. 5. LSVN simulation: the average distance around the swarm center for the Rastrigin's problem

of 95%, the Friedman test suggests rejecting the null hypothesis H_0 (p-value=0.00 < 0.05), which means that there exist highly significant differences between at least two methods across benchmark. But how can these algorithms be determined?

At the second step it is necessary to prove that the proposed algorithm is involved in the highly significant differences reported by the Friedman test. To do that, we compute the Wilcoxon signed rank test [40] with the intention to identify significant difference between pairs of algorithms. The Wilcoxon test attempts to answer a simple question: do two samples represent two different populations? Table 4 shows the p-values computed by this procedure. Using a significance level of 0.05, corresponding to the 95% confidence interval, the Wilcoxon test suggests rejecting the null hypothesis H_0 (p-value < 0.05) for pairs involving the new algorithm PSO-LSVN. It definitively confirms that there exist highly significant differences between POS-RSVN and others algorithms, where the new algorithm is the winner since PSO-RSVN has the lowest mean rank according to Table 3.

In addition, in experiments reported in Table 4 we included others algorithms reporting good performance such as the original PSO-RSVN, while remaining pair-wise comparisons were omitted for better analysis of statistical outcomes. Also, the omitted pair-wise samples do not suggest any relevant information demonstrating the comparative superiority of our proposal. Then, from computed results a relevant conclusion came out: the Wilcoxon test reveals that there exist no enough reasons for rejecting the null hypothesis H_0 (which means that p-value > 0.05) for the following pairs: PSO-RSVN vs. BBPSO, PSO-RSVN vs. CLPSO, and PSO-RSVN vs. Lbest. However, the simplicity and robustness of BBPSO make this approach a preferable choice for handling problems with a large number of multimodal optima.

In general, the PSO-LSVN method introduces three new parameters that should be fixed by the experts: the threshold α for the average distance around the swarm center, the number of evaluations of the objective function without progress, and the number of neighborhoods M. The first factor is a real-value used as a low threshold for the diversity measure. The wrong selection of this parameter could be significant for the algorithm performance: too high values will affect the PSO-LSVN exploitation ability, due to the fact that false premature convergence states could be detected, whereas too low values could never detect an existing premature convergence state. Empirical results showed that values from 1.0E-2 to 1.0E-8 are a good choice.

Another central user-specify parameter is the allowed number of evaluations without progress; it is relatively easy to set and will be automatically calculated depending on the maximal number of evaluations for the algorithm execution (e.g. $P = n/10$). The reader should notice that it will be used if the swarm diversity remains stable but the objective function does not suffer any improvement. As a final point, the number of neighborhoods controls the partitions' number used in the sampling stage. The recommended values for this parameter are either $M = 5$ or $M = 10$. In fact, it could be successfully fixed to $M = 10$ in all the cases.

For better understating of the behavior of the swarm reorganization process, Fig. 5 illustrates the average distance around the swarm center for the Shifted Rastrigin's Function during the optimization process. It is a multimodal, shifted, separable benchmark having a huge number of local optima. Of course this problem could be a challenge for any optimization problem, since a large number of local optima could induce premature convergence states. In fact, it should be perceived that the swarm diversity is gradually lost, but it is restored using the reorganization scheme implemented in the PSO-LSVN algorithm.

Of course, only guarantying proper swarm diversity we can't ensure to avoid local optima. But preserving the global diversity in the artificial population helps PSO-based algorithms to discover better peaks, hence enhancing the metaheuristic performance. Moreover, most attempts to do that perform quite well for multimodal problems, but the global convergence rate is seriously affected when unimodal or even relatively simple multimodal functions are optimized. Actually, another conclusion from experiments is that the proposed PSO-LSVN is capable to efficiently optimize both unimodal and multimodal benchmark.

As a future work the authors will be enrolled in a deeper study of the effects on varying the parameters over the algorithm behavior. Besides, it is relevant to survey the algorithm performance when the dimensionality of the search space increases, and hence increases the total number of local optima. To finish this section, we present the reader an open question: could the basis of the LSVN procedure be a starting point to formulate a new metaheuristic?

7 Conclusions

Particle Swarm Optimization has proven to be an effective search method for solving challenging real-valued optimization problems. However, PSO algorithms using a *gbest* model or fully connected topology frequently are attracted to local solutions causing premature convergence, as a result of the progressive lost of the population diversity. Also, stagnation configurations are frequent, where particles are unable of discovering better optima, but the particle swarm diversity remains "stable" during the algorithm execution. To deal with these undesirable drawbacks, several approaches have been introduced, mainly oriented to increase the diversification in the artificial population.

In this paper we proposed a modification of the RSVN procedure, which was designed to deal with non-progress configurations. Here the key idea of the RSVN procedure is to improve the diversity of the artificial population from the selection of samples around the global-best point found during the search.

The modifications introduced in this paper are mainly oriented to replace the uniform distribution by a Lévy generator, and to incorporate a novel operator for recombining the selected samples. Summarizing, these variations allows the PSO-LSVN algorithm to perform better exploration during the sampling process, thus leading to a reasonably improved particle swarm.

We also conducted statistical analyses which confirm that there exist highly significant differences between the proposed PSO-LSVN and the other algorithms selected for comparison. From our point of view these results are a direct consequence of the strategy for detecting non-progress configurations, as well as the scheme for reorganizing the swarm.

It should be also remarked that the LSVN procedure induces a reduced extra-computational cost since it is activated in some iterations thus preserving the high convergence rate of the PSO metaheuristic. Although LSVN requires specifying three parameters that may be easily fixed by users, future work will be focused on extending the statistical analysis to the effects of parameters on the PSO-LSVN performance. As well, its behavior when the dimensionality of the search space increases will be studied across benchmark.

References

1. **Kennedy, J. and Eberhart, R. (1995).** Particle Swarm Optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, 1942—1948.
2. **Eberhart, R. & Kennedy, J. (1995).** A New Optimizer using Particle Swarm Theory. *Sixth International Symposium on Micromachine and Human Science*, Nagoya, Japan, 39–43.
3. **Bratton, D. & Kennedy, J. (2007).** Defining a Standard for Particle Swarm Optimization. *IEEE Swarm Intelligence Symposium (SIS 2007)*, Honolulu, Hawai, 120–127.
4. **Wang, Y., Li, B., Weise, T., Wuang, J., Yuan, B., & Tian, Q. (2011).** Self-adaptive learning based particle swarm optimization. *Information Sciences: an International Journal*, 181(20), 4515–4538.
5. **Kennedy, J., Russell, C.E., & Shi, Y. (2001).** Swarm Intelligence. San Francisco: Morgan Kaufmann Publishers.
6. **Evers, G.I. & Ben, M. (2009).** Regrouping Particle Swarm Optimization: A new Global Optimization Algorithm with Improved Performance Consistency across Benchmarks. *IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, TX, 3901–3908.
7. **Liu, Y., Ling, X., Shi, Z., Mingwei, L.V., Fang, Ji., & Zhang, L. (2011).** A survey on Particle Swarm Optimization algorithms for multimodal function optimization. *Journal of Software*, 6(12), 2449–2455.
8. **Chen, S. & Montgomery, J. (2011).** A simple strategy to maintain diversity and reduce crowding in Particle Swarm Optimization. AI 2011: Advances in Artificial Intelligence. *Lecture Notes in Computer Science*, 7106, 281–290.
9. **Kennedy, J. & Mendes, R. (2006).** Neighborhood topologies in fully informed and best of neighborhood particle swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 36(4), 515–519.
10. **Nápoles, G., Grau, I., & Bello, R. (2012).** Particle Swarm Optimization with Random Sampling in Variable Neighbourhoods for solving Global Minimization Problems. *Swarm Intelligence, Lecture Notes in Computer Science,* 7461, 352–353.
11. **Nápoles, G., Grau, I., & Bello, R. (2012).** Constricted Particle Swarm Optimization based algorithm for global optimization. *POLIBITS,* 46, 5–11.
12. **Clerc, M. & Kennedy, J. (2002).** The particle swarm -explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1), 58–73.
13. **Poli, R., Kennedy, J., & Blackwell, T. (2007).** Particle Swarm Optimization. *Swarm Intelligence*, 1(1), 37–57.
14. **Shi, Y. & Eberhart, R. (1998).** A Modified Particle Swarm Optimizer. *1998 IEEE International Conference on Evolutionary Computation Proceedings*, Anchorage, AK, 69–73.
15. **Trelea, I.C. (2003).** The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6), 317–325.
16. **Kennedy, J. (2003).** Bare bones particle swarms. *2003 IEEE Swarm Intelligence Symposium*, Indianapolis, USA, 80–87.
17. **Richer, T.J. & Blackwell, T.M. (2006).** The Lévy particle swarm. *2006 IEEE Congress on Evolutionary Computation*, Vancouver, BC, Canada, 808–815.
18. **Kennedy, J. (2004).** Probability and dynamics in the particle swarm. *IEEE Congress on Evolutionary Computation (CEC 2004)*, Portland, OR, USA, 1, 340–347.
19. **Liang, J.J., Qin, A.K., Suganthan, P.N., & Baskar, S. (2006).** Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Transactions on Evolutionary Computation*, 10(3), 281–295.
20. **Wang, H., Li, H., Liu, Y., Li, C., & Zeng, S. (2007).** Opposition-based Particle Swarm Algorithm with Cauchy mutation. *IEEE Congress on Evolutionary Computation*, Singapore, 4750–4756.
21. **Tizhoosh, H.R. (2006).** Opposition-based reinforcement learning. *Journal of Advanced Computational Intelligence and Intelligent and Intelligent Informatics*, 10(4), 578–585.
22. **Riget, J. & Vesterstrøm, J.S.(2002).** *A Diversity-Guided Particle Swarm Optimizer – the ARPSO* (Technical Report no. 2002-02). Aarhus C, Denmark: Aarhus Universitet.
23. **Pant, M. & Thangaraj, R. (2007).** A new Particle Swarm Optimization with quadratic crossover. *International Conference on Advanced Computing and Communications*, Guwahati, Assam, 81–86.
24. **Zan, Z.H., Zhang, J., Li, Y., & Chung, H.S.H. (2009).** Adaptive Particle Swarm Optimization. *IEEE Transactions on Systems, Man and*

Cybernetics - Part B: Cybernetics, 39(6), 1362–1381.

25. **Olorunda, O. & Engelbrecht, A.P. (2008).** Measuring Exploration/Exploitation in Particle Swarms using swarm diversity. *IEEE Congress on Evolutionary Computation*, Hong Kong, China, 1128–1134.
26. **Suganthan, P.N., et al. (2005).** Problem definition and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *Technical Report 2005005*
27. **Kirkpatrick, S., Gellat, C.D., & Vecchi, M.P. (1983).** Optimization by simulated annealing. *Science*, 220(4598), 671–680.
28. **Nápoles, G., Grau, I., León, M., & Grau, R. (2013).** Modelling, aggregation and simulation of a dynamic biological system through Fuzzy Cognitive Maps. *Advances in Computational Intelligence, Lecture Notes in Computer Science,* 7630, 188–199.
29. **Storn, R. & Price, K. (1997).** Differential Evolution - A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4), 341–359.
30. **Pavlyukevich, I. (2007).** Lévy flights, non-local search and simulated annealing. *Journal of Computational Physics*, 226(2), 1830–1844.
31. **Yang, X.S. (2008).** Nature-Inspired Metaheuristic Algorithms. United Kingdom: Luniver Press.
32. **Lee, C.Y. & Yao, X. (2004).** Evolutionary programming using mutations based on the Lévy probability distribution. *IEEE Transactions on Evolutionary Computation*, 8(1), 1–13.
33. **Lévy, P. (1937).** Theorie de l'addition des Veriables Aleatoires, Paris: Guathier-Villars.
34. **Gnedenko, B.V. & Kolmogorov, A.N. (1954).** Limit Distribution for Sums of Independent Random Variables, Michigan: Addison-Wesley.
35. **Mantegna, R.N. (1994).** Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes. *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*, 49(5), 4677–4683.
36. **Higgins, J.J. (2003).** *Introduction to Modern Nonparametric Statistics.* Pacific Grove, CA : Brooks/Cole.
37. **García, S., Fernández, A., Luengo, J., & Herrera, F. (2009).** A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability. *Soft Computing*, 13(10), 959–977.
38. **García, S., Molina, D., Lozano, M., & Herrera, F. (2009).** A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: A case study on the CEC'2005 special session on real parameter optimization. *Journal of Heurisitcs*, 15(6), 617–644.
39. **Friedman, M. (1937).** The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200), 675–701.
40. **Wilcoxon, F. (1945).** Individual comparisons by ranking methods. *Biometrics Bulletin,* 1(6), 80–83.

Gonzalo Nápoles received the B.Sc. degree (with honors) in Computer Sciences from the Las Villas Central University "Marta Abreu" (UCLV), Cuba, in 2011. He has authored/coauthored two monographs and 15 papers in conference proceedings and scientific journals. He earned the Cuban National Award for Computer Science Students twice (2010 and 2011), the Academy of Sciences Award in 2012, and the Best Paper Award Third Place at MICAI 2012 conference. His research interests include soft computing, fuzzy cognitive maps, metaheuristics, evolutionary computation, machine learning and knowledge discovering.

Isel Grau received the B.Sc. degree (with honors) in Computer Sciences from the Las Villas Central University "Marta Abreu" (UCLV), Cuba, in 2011. She has authored/coauthored 10 papers in conference proceedings and scientific journals. She earned the Cuban National Award for Computer Science Students in 2010, the Cuban National Award for Best Diploma Thesis in 2011, the Cuban Academy of Sciences Award twice (2011 and 2012) and the Best Paper Award Third Place at MICAI 2012 conference. Her research interests include soft computing, bioinformatics, recurrent neural networks, statistics and machine learning.

Marilyn Bello received the B.Sc. degree (with honors) in Computer Sciences from the Las Villas Central University "Marta Abreu" (UCLV), Cuba, in 2012. She has authored/coauthored several papers in conference proceedings and scientific journals. She obtained several awards in different student scientific events. Her research interests include metaheuristics and machine learning.

Rafael Bello received his Bachelor degree in Mathematics and Computer Science (1982) from the Las Villas Central University "Marta Abreu", Santa Clara, Cuba, and his Ph.D. in Mathematics from UCLV in 1988. He has been a visiting scholar at some universities in Spain, Germany and Belgium. He is a Full Professor at the Computer Science Department, UCLV, Cuba, and exhibits a long record of academic exchange with many universities in Latin America and Europe. He has taught more than 45 undergraduate and graduate courses in those academic centers. He has authored/edited 9 books, published over 180 papers in conference proceedings and scientific journals, supervised over 30 Bachelor, Master and Ph.D. thesis. He has deserved prestigious awards from the Cuban Academy of Sciences and other renowned scientific societies. His research interests comprise soft computing (rough and fuzzy set theories), metaheuristics, machine learning and decision making.

Article received on 05/05/2013, accepted on 08/07/2013.