10th Brazilian Congress on Computational Intelligence (CBIC'2011), November 8 to 11, 2011, Fortaleza, Ceará, Brazil
© Brazilian Society on Computational Intelligence (SBIC)

A Cooperative Approach using Particle Swarm Optimization with Adaptive Behavior

L. N. Vitorino, S. F. Ribeiro and C. J. A. Bastos-Filho
University of Pernambuco
{lnv,sfr}@ecomp.poli.br, carmelofilho@ieee.org

Abstract – Particle Swarm Optimization (PSO) has been widely used to solve real world optimization problems. Since the original PSO fails to maintain the diversity along the search process, some PSO variations have been proposed to overcome this limitation. The recently introduced Adaptive Particle Swarm Optimization (APSO) presents faster convergence, while maintaing the population diversity, by including into the PSO algorithm the auto-adaptation of the parameters according to the current spatial distribution of the swarm. Besides, there are some PSO variations which incorporate a cooperative behavior, such as the Clan Particle Swarm Optimization (ClanPSO). In this paper, we propose to include the auto-adaptation ability solely in the Clans of the ClanPSO algorithm and compare its performance to previous approaches. We evaluated our proposal in six benchmark functions recently proposed in *IEEE Congress on Evolutionary Computation 2010.* Our proposal obtained similar or better results in terms of fitness when compared to the original ClanAPSO, but presents a lower computational cost. We obtained better results when compared to other cooperative approaches and the classical PSO approaches.

Keywords – Swarm Intelligence, Particle Swarm Optimization, Cooperative Optimization, Auto-adaptation.

1. INTRODUCTION

Particle Swarm Optimization (PSO) [1] is a population based algorithm inspired by the social behavior of flocks of birds. The optimization in PSO occurs balancing the social learning and the individual learning of the individuals within the swarm. The cooperation and competition among the particles plays an important role in the exploration of the search space [2].

Each particle has four main attributes: the position in the search space $\vec{x}_i(t)$, the velocity in the search space $\vec{v}_i(t)$, the best position found by the particle during the search process (cognitive memory, $\vec{p}_i(t)$), and the best position found by the neighborhood of the particle (social memory, $\vec{n}_i(t)$). A neighborhood is the set of particles of the swarm from which a particle is able to acquire information.

The particles of the swarm move through the search space by updating their velocities and positions [3]. All particles are updated in all iterations of the algorithm. There are several equations that can be used to update the velocity of the particles. In this paper, we used the equation proposed in [4] and depicted in (1) because it can avoid explosion states.

$$\vec{v}_i(t+1) = \chi\{\vec{v}_i(t) + c_1 r_1[\vec{p}_i(t) - \vec{x}_i(t)) + c_2 r_2[\vec{n}_i(t) - \vec{x}_i(t)]\}, \tag{1}$$

where

$$\chi = \frac{2\kappa}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \tag{2}$$

and

$$\varphi = c_1 r_1 + c_2 r_2, \tag{3}$$

where c_1 and c_2 are called acceleration constants, r_1 and r_2 are two random numbers generated by an uniform probability distribution function $U(0,1)$. The constriction factor χ assumes a value in the interval $[0,1]$. The parameter κ controls the exploration-exploitation ability of the swarm.

The goals to improve the PSO algorithm are to accelerate the convergence and avoid to be trapped into local optima. Since these goals are often conflicting, many PSO variations have been proposed aiming to overcome both problems simultaneously [5] [6] [7] [8].

Some approaches have been proposed to dynamically adapt the acceleration constants and the inertia factor during the search process [9] [10]. Zhan *et al.* [10] introduced the Adaptive Particle Swarm Optimization (APSO) in 2009, where the state of convergence of the swarm is estimated at each iteration and the PSO parameters are updated based on this state.

On the other hand, many communication topologies to define the flow of information among the particles were proposed in the last decade. These topologies aims to avoid premature convergence, while maintaing the convergence speed. The Clan Particle Swarm Optimization (ClanPSO) [11] is a dynamic topology based on sub-swarms, called Clans, which presented interesting results when compared to other topologies. In 2011, Pontes *et al.* [12] proposed to include the adaptation ability in the the update procedures of the Clans and in the conference of leaders of the ClanPSO algorithm. In this paper, we propose to include the auto-adaptation ability solely in the Clans of the ClanPSO algorithm and compare its performance to the previous approaches in a recently proposed set of benchmark functions.

This paper is organized as follows. In section 2, we present the Adaptive Particle Swarm Optimization. In section 3, we present the basic concepts about Clan Particle Swarm Optimization. In the section 4, we describe our proposal to include the adaptation capability into the Clans. In section 5, we present the simulation setup and the results for benchmark functions presented in the *IEEE Congress on Evolutionary Computation 2010.* Finally, we present our conclusions in section 6.

2. ADAPTIVE PARTICLE SWARM OPTIMIZATION

Zhan *et al.* [10] identified some deficiencies in the PSO behavior concerning the control of the velocity and the capacity to escape from local minima. Because of this, Zhan *et al.* proposed the Adaptive PSO (APSO), which presents a systematic scheme to update the PSO parameters based on fuzzy rules. APSO can be viewed as a sequence of iterations of the following steps: Estimate and classify the evolutionary state (f_{evol}), determine the acceleration coefficients based on the current value of f_{evol}, learn using elitism and adapt the inertia factor. These steps are described in the next subsections.

2.1 Estimating the Evolutionary State

The first step is to estimate the evolutionary state of the swarm. It can be inferred by evaluating the evolutionary factor f_{evol}. It is evaluated in two steps. At first, one has to calculate the mean distance of each particle to the others by using (4). After this, one has to evaluate f_{evol} by using (5).

$$d_i = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} \sqrt{\sum_{k=1}^{D} (x_i^k - x_j^k)}, \tag{4}$$

where N is the number of particles in the swarm and D is the number of dimensions of the particles.

$$f_{evol} = \frac{d_g - d_{min}}{d_{max} - d_{min}} \ \epsilon \ [0, 1], \tag{5}$$

where d_g is the average distance between the best particle of the swarm and the other particles, d_{min} and d_{max} and the smallest and the biggest average distance among all particles, respectively.

2.2 Classifying the Evolutionary State

The state of convergence of the swarm is classified by fuzzy rules, where the evolutionary state with the higher value for the membership function is chosen at each iteration. Figure 1 presents the membership functions used by APSO. The states of convergence are: *Convergence state* – Occurs for very low values of f_{evol}. It means that all particles are probably located at the same local minimum; *Exploitation state* – Occurs for low values of f_{evol}. It means that average distance between the best particle of the swarm to the other particles is small; *Exploration state* – Occurs for medium values of f_{evol}. It means that the average distance between the best particle of the swarm and the other particles is not too small, but probably none of the particles are located in a region far from the position of the best particle. In this case, the particles might be exploring the search space at large and are not exploiting just one region of the search space; *Jumping out*: Occurs for high values of f_{evol}. It means that one or more particles are far from the best particle. It can occur when a particle escapes from a local minimum.

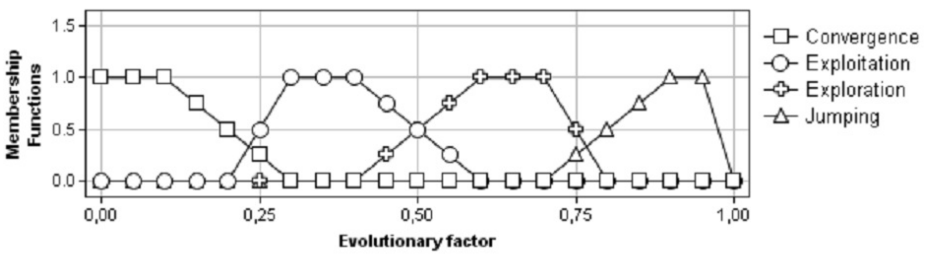

Figura 1: Fuzzy membership functions for the four evolutionary states.

2.3 Determining the acceleration coefficients

The acceleration coefficients are initialized with the values equal to 2.0 and are updated according to the current value of f_{evol}. The rules to update the coefficients are depicted in Table 1. In order to avoid extreme values for the acceleration coefficients, it is necessary to normalize them. We used a minimum and maximum values equal to c_{min} = 1.5 and c_{max} = 2.5, respectively.

Table 1: Rules to update accelerate coefficients according to the current evolutionary state.

Evolutionary State	c_1	c_2
Exploration	increment	decrement
Exploitation	increment slightly	decrement slightly
Convergence	increment slightly	increment slightly
Escape	decrement	increment

2.4 Learning Strategy using Elitism

This strategy aims to improve the exploration ability of the algorithm. It is applied only in one dimension of the better particle of the swarm in order to allow the swarm to escape from a local optimum. The operator is applied as a Gaussian mutation according to (6), where all the dimensions have the same probability to be chosen. The position of the best particle is just updated if the new position found by the operator is better than the previous one.

$$x_{pbest_d}(t+1) = x_{pbest_d}(t) + (X^d_{max} - X^d_{min})G(\mu, \sigma^2), \tag{6}$$

where (X^d_{max}, X^d_{min}) are the search space boundaries in the d^{th} dimension, $G(\mu, \sigma^2)$ is a random number generated by a normal distribution $N(\mu, \sigma)$ and σ is the elitism learning rate. Zhan *et al.* [10] proposed to use σ = 1.0 in the beginning of the simulations and to decrease it to σ = 0.1 at the end of the simulation.

2.5 Adaptation of the Inertia Factor

The inertial factor ω can be used to adjust the exploration-exploitation ability of the swarm. ω is evaluated as function of f_{evol} by using the equation (7).

$$\omega(f) = \frac{1}{1 + 1.5e^{-2.6 f_{evol}}} \ \epsilon \ [0.4; 0.9], \ \ \forall f \ \epsilon \ [0,1]. \tag{7}$$

3. CLAN PARTICLE SWARM OPTIMIZATION

ClanPSO [11] consists in a set of clans, where each clan is considered as a sub-swarm and uses a fully-connected structure to share information within the sub-swarm. For each iteration, each one of clans performs a search using a standard PSO and selects the particle with the best information of the entire clan. This particle is assigned as the leader. The delegation process is performed with the G_{best} approach.

After the delegation, the leaders adjust their positions running a PSO execution solely with the leaders. This process is called conference of leaders. The conference can be performed using either the G_{best} or L_{best} topology. If the conference of leaders is performed using the G_{best} approach (called ClanPSO-G), the exploitation capacity is improved. Otherwise, if the L_{best} approach (called ClanPSO-L) is used for conference of leaders, the exploration capacity is improved.

After the conference of leaders, the leaders return to their respective clans and the new information acquired during the conference is used to update the other particles of the clan. Indirectly, it allows the other particles to be guided by the best position found within the entire topology. However, one must observe that a foreign leader does not directly influence another clan. This mechanism helps to preserve the exploration capacity of the clans.

4. THE ADAPTIVE BEHAVIOR IN CLAN PARTICLE SWARM OPTIMIZATION

In this paper, we propose to include the adaptation capability in the PSO execution within the clans. We call this approach *Clan Adaptive Particle Swarm Optimization* (ClanAPSO). Pontes *et al.* [12] proposed previously to run the APSO algorithm within the clans and also execute the APSO in the conference of leaders. However, we perceived from simulations that the use of the APSO in the conference of leaders returns similar results if one runs a simple PSO in the conference of leaders, but needs more time to be executed. We believe that by using ClanAPSO, each clan can adapt itself independently and avoid an excessive exploitation of the same region of the search space. The pseudocode of our proposal is shown in Algorithm 1.

5. SIMULATION SETUP

We performed our simulations in six benchmark minimization problems recently proposed in [14]. $F2$ (Shifted Rastrigin's Function), $F10$ (*D/2m*-group Shifted and *m*-rotated Rastrigin's Function) and $F16$ (*D/m*-group Shifted and *m*-rotated Ackley's Function) are multimodal functions, while $F4$ (*Single*-group Shifted and *m*-rotated Elliptic Function), $F7$ (*Single*-group Shifted *m*-dimensional Schwefel's Problem 1.2) and $F19$ (Shifted Schwefel's Problem 1.2) are unimodal functions. Table 2 shows the search space and the global minimum for each benchmark function. We selected these function in order to represent all classes of problems described in the entire benchmark. All simulations were performed using 30 particles and 100 dimensions for all benchmark functions (except in the subsection 6.3). All results were obtained over 30 trials.

Algorithm 1: Pseudocode of the algorithm ClanAPSO.

```
Initialize the swarm;
while the stop criterion is not achieved do
    for each clan do
        execute APSO within the Clan;
        delegate leader;
    end
    create conference of leaders;
    execute PSO with the leaders;
end
```

Table 2: Benchmark functions, search space boundaries and global optimum position.

Function	Search space	Opt
F2	$-5 \leq x_i \leq 5$	0.0^D
F4	$-100 \leq x_i \leq 100$	0.0^D
F7	$-100 \leq x_i \leq 100$	0.0^D
F10	$-5 \leq x_i \leq 5$	0.0^D
F16	$-32 \leq x_i \leq 32$	0.0^D
F19	$-100 \leq x_i \leq 100$	0.0^D

6 SIMULATION RESULTS

In order to analyze different issues and behaviors generated by the inclusion of the adaptation ability in the clans, we decided to separate the results according to three aspects. These aspects are: Influence of the number of clans, convergence analysis and dependence with the dimensionality.

6.1 Influence of the Number of Clans

As the number of clans plays an important role on the performance of the ClanAPSO, one should analyze the behavior of the algorithm ClanAPSO using different configurations. In order to maintain the total number of particles equal to 30, we analyzed three different arrangements: 3 Clans with 10 particles (3x10), 6 Clans with 5 particles (6x5) and 10 Clans with 3 particles (10x3). For comparison sake, we simulated the same situations for the original ClanPSO. The algorithms were ran for 1,000 iterations.

Figures 2, 3 and 4 present the results for the algorithms ClanAPSO and ClanPSO for the unimodal functions. One can observe that all the ClanAPSO configurations outperformed the ClanPSO configurations. In general, the conference of leaders with G_{best} achieved slightly better results. Furthermore, in the studied range, one can notice that number of clans do not influence a lot on the performance of the algorithm.

6.2 Convergence Analysis

We also performed a comparison of various algorithms evolving along the iterations to analyze the convergence speed. We run $1,000$ iterations for all benchmark functions. We did not observe a significative improvement in the convergence speed for unimodal functions. However, Figures 5, 6 and 7 depict the evolution of PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 with L_{best} topology for the multimodal benchmark functions. One can notice that the ClanAPSO far outperforms the other algorithms.

6.3 Analysis of the Dependence on the Dimensionality of the Problem

This subsection aims to analyze the scalability of the algorithms. As the dimensionality increases, the difficulty of the problem also increases. We did not observe any significant improvement for the $F16$ function, but the difference in the performance between the ClanAPSO and the other approaches is maintained as the dimensionality of the problem also increases. Figures 8, 9 and 10 shows the fitness as a function of the dimensionality for $F2$, $F4$ and $F7$ benchmark functions.

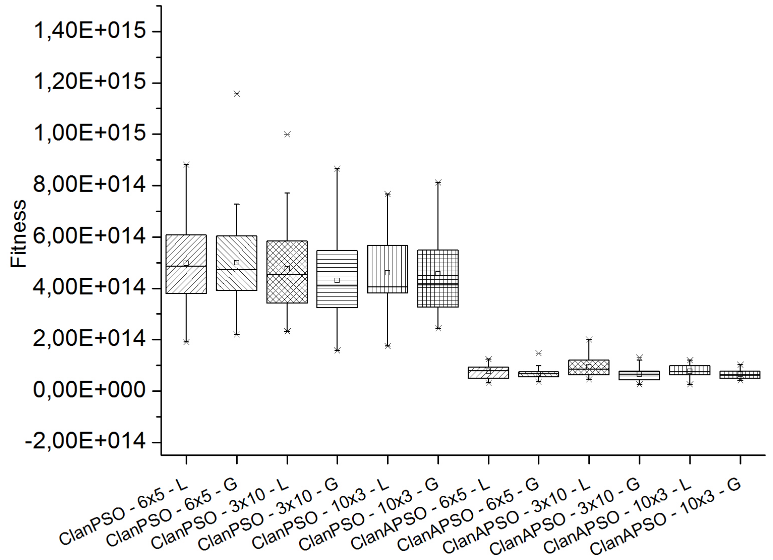

Figure 2: Boxplot of the fitness for the $F4$ function varying the number of clans.

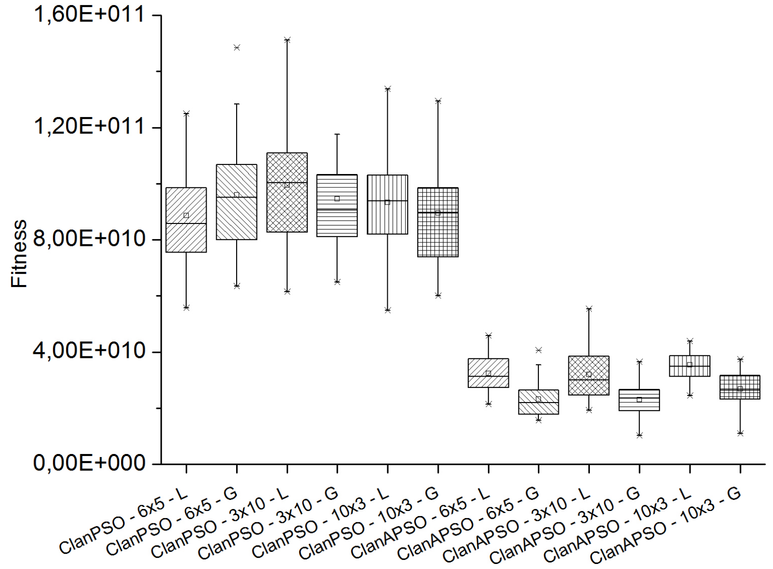

Figure 3: Boxplot of the fitness for the $F7$ function varying the number of clans.

7. CONCLUSIONS

In this paper we included the adaptation capability in a cooperative particle swarm optimization approach. We showed that different clans can execute simultaneously in parallel the APSO algorithm and the conference of leaders can run a simple PSO, without any performance degradation. We showed that adaptation ability allowed the algorithm to search deeply in several local minima, while the cooperative approach intensified the search around the global optimum. When comparing our proposal to other approaches, we conclude that the ClanAPSO is more effective in multimodal search spaces. Besides, we observed that the number of clans in not a critical aspect.

REFERENCES

[1] R. Eberhart and J. Kennedy. "A New Optimizer Using Particle Swarm Theory". In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–45. IEEE Service Center, 1995.

[2] A. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*. Wiley, Pretoria, 2006.

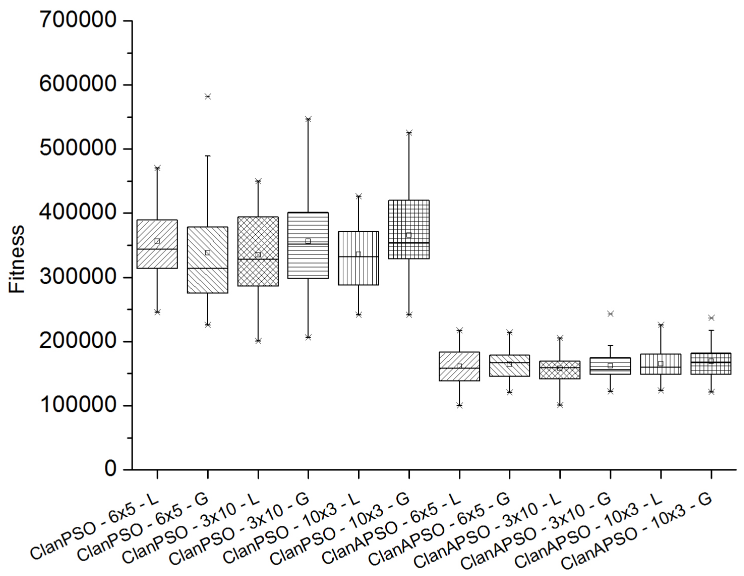

Figure 4: Boxplot of the fitness for the $F19$ function varying the number of clans.

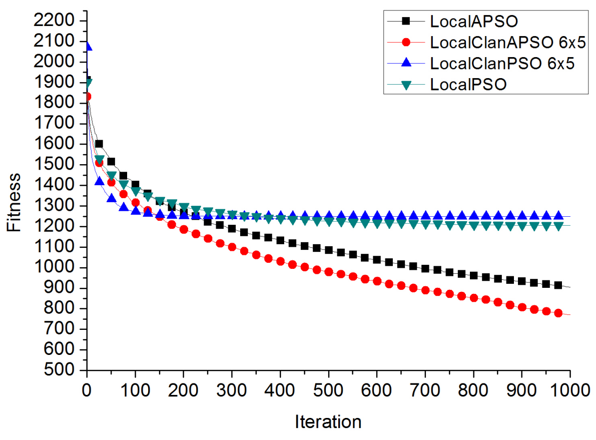

Figure 5: Fitness function evolution of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 for the 100-D $F2$ function.

[3] J. Kennedy. "Why does it Need Velocity?" In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 38–44. IEEE Computer Society, 2005.

[4] M. Clerc and J. Kennedy. "The Particle Swarm-explosion, Stability, and Convergence in a Multidimensional Complex Space". *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[5] S. Y. Ho, H. S. Lin, W. H. Liauh and S. J. Ho. "OPSO: Orthogonal Particle Swarm Optimization and its Application to Task Assignment Problems". *IEEE Transactions on Sytems, Man and Cybernetics, Part A: IEEE Transactions on Systems and Humans*, vol. 38, no. 2, pp. 288–298, 2008.

[6] B. Liu, L. Wang and Y. H. Jin. "An Effective PSO Based Memetic Algorithm for Flow Shop Scheduling". *IEEE Transactions on Systems, Man and Cybernetics*, vol. 37, no. 1, pp. 18–27, 2007.

[7] G. Ciuprina, D. Ioan and I. Munteanu. "Use of Intelligent-particle Swarm Optimization in Eletromagnetics". *IEEE Transactions on Magnetics*, vol. 38, no. 2, pp. 1037–1040, 2002.

[8] J. J. Liang, A. K. Qin, P. N. Suganthan and S. Baskar. "Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions". *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

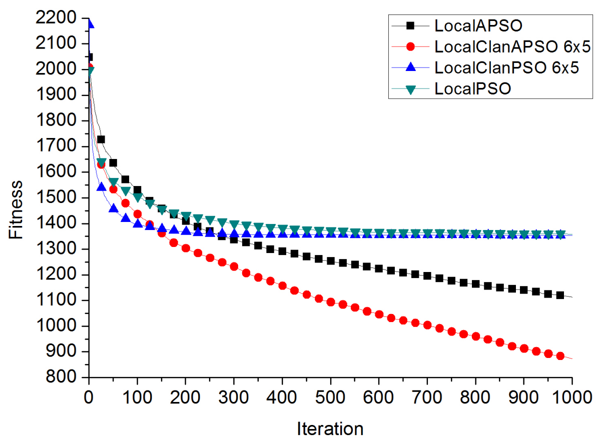

Figure 6: Fitness function evolution of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 for the 100-D $F10$ function.

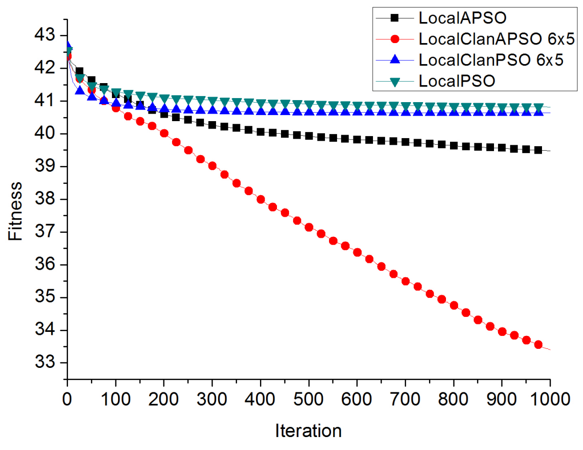

Figure 7: Fitness function evolution of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 for the 100-D $F16$ function.

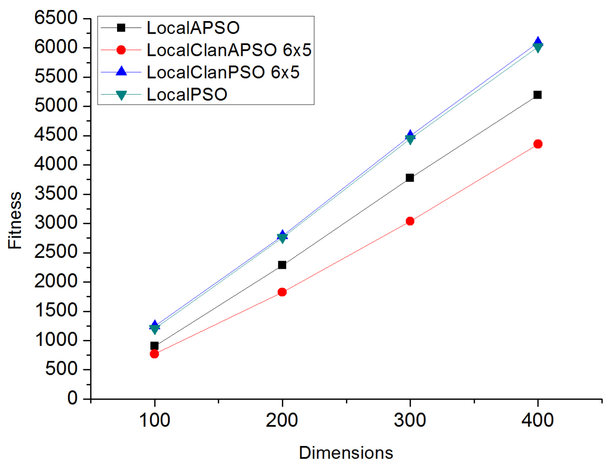

Figure 8: Performance of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 as a function of the dimensionality for $F2$ function.

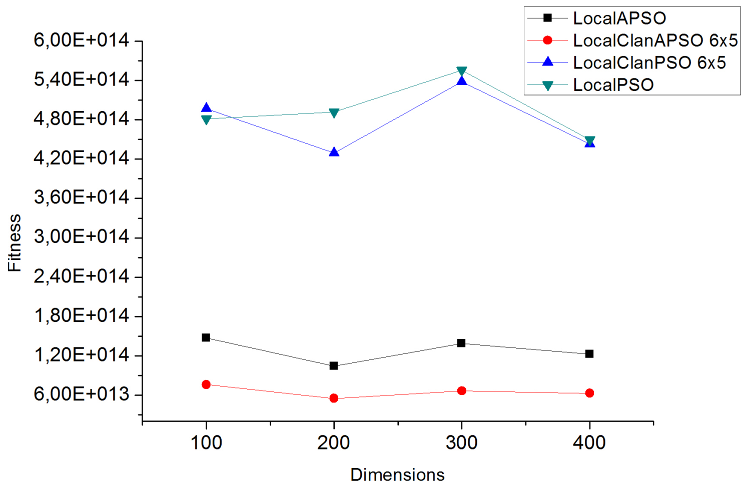

Figure 9: Performance of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 as a function of the dimensionality for $F4$ function.

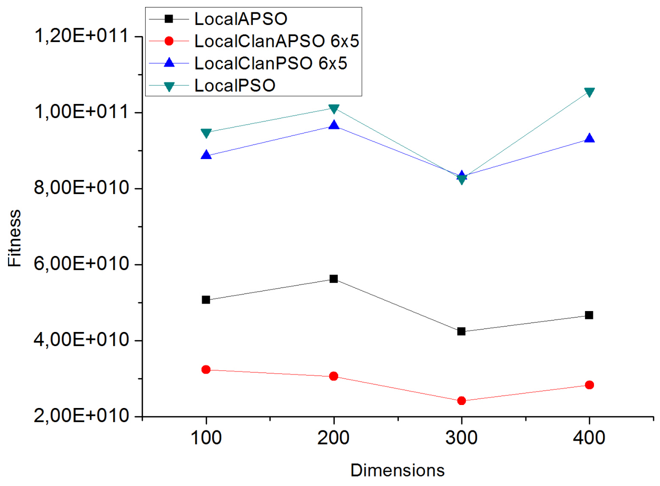

Figure 10: Performance of the PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 as a function of the dimensionality for $F7$ function.

[9] A. Ratnaweera, S. Halgamuge and H. Watson. "Self-organizing Hierarchical Particle Swarm Optimizer with Time-varing Acceleration Coefficients". *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[10] Z. Zhan, J. Zhang, Y. Li and H. Chung. "Adaptive Particle Swarm Optimization". *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.

[11] D. F. Carvalho and C. J. A. Bastos-Filho. "Clan Particle Swarm Optimization". *IEEE Congress on Evolutionary Computation*, pp. 3044–3051, 2008.

[12] M. R. Pontes, F. B. Lima-Neto and C. J. A. Bastos-Filho. "Adaptive Clan Particle Swarm Optimization". In *Proceedings of IEEE Swarm Intelligence Symposium*, pp. 17–22. IEEE Computer Society, 2011.

[13] C. J. A. Bastos-Filho, D. F. Carvalho, E. M. N. Figueiredo and P. B. C. de Miranda. "Dynamic Clan Particle Swarm Optimization". *Ninth International Conference on Intelligent Systems and Applications - ISDA2009*, pp. 249–254, 2009.

[14] K. Tang, X. Li, P. N. Suganthan, Z. Yang and T. Weise. "Benchmark Functions of the CEC'2010 Special Session and Competition on Large Scale Global Optimization". Technical report, University of Science and Technology of China (USTC), School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL): Héféi, Ānhuī, China, 2009.