

Adaptive Clan Particle Swarm Optimization

Murilo R. Pontes, Fernando B. Lima Neto, Carmelo J. A. Bastos-Filho
Polytechnic School of Pernambuco
University of Pernambuco
Recife, Pernambuco 50.720-001
Email: carmelofilho@ieee.org

Abstract—Particle Swarm Optimization has been widely used to solve real world problems, mainly when there are too many variables to be optimized and these variables are continuous. In nature one can observe many examples of cooperative behaviors that lead to complex problem solving. Recently, some Particle Swarm Optimization variations gracefully incorporate such cooperative features with consequent beneficial new abilities. In this paper we put forward the incorporation of auto-adaptation capability in a cooperative Particle Swarm Optimization algorithm, called Clan Particle Swarm Optimization. Next, we present a deep analysis on the adaptation process for one multimodal function and evaluate the performance of our proposal in some well known benchmark problems. The results revealed that our proposal achieved better performance than other approaches, specially in tough multimodal problems.

Index Terms—Particle Swarm Optimization; Cooperative Optimization; Adaptive Systems; Swarm Intelligence.

I. INTRODUCTION

The Particle Swarm Optimization Algorithm (PSO) was proposed by Kennedy and Eberhart in 1995 [1] [2]. PSO is a population based algorithm where each particle has four attributes: the position in the search space $\vec{x}_i(t)$, the velocity within 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 acquired by the particles of its neighborhood during the search process (social memory) $\vec{n}_i(t)$. The position of a particle represents a possible solution for the fitness function. At each iteration, the velocity and the position of all the particles are updated. The cognitive and the social memory are just updated if the new position outperforms the stored values. The velocity of the particles are updated according to a velocity equation (1), which in this paper is the widely known and used proposal developed by Clerc [3]. The first term in the right side of equation (1) takes into account the current velocity assumed by the particle and prevents drastic changes of directions. The second and the third terms are called cognitive and social components, respectively.

$$\begin{aligned} \vec{v}_i(t+1) &= \chi[\vec{v}_i(t) + c_1\epsilon_1(\vec{p}_i(t) - \vec{x}_i(t)) \\ &+ c_2\epsilon_2(\vec{n}_i(t) - \vec{x}_i(t))], \end{aligned} \quad (1)$$

where

$$\chi = \frac{2\kappa}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \varphi = c_1\epsilon_1 + c_2\epsilon_2, \quad (2)$$

where c_1 and c_2 are called acceleration constants, ϵ_1 and ϵ_2 are two random numbers generated by using an uniform prob-

ability distribution function $U(0, 1)$. The constriction factor χ assumes a value between 0 and 1. In the case where $\varphi \geq 4$ and $\chi \in [0, 1]$, explosion states are avoided. The parameter κ , in equation (2), controls the exploration-exploitation tradeoff of the swarm.

In order to determine $\vec{n}_i(t)$, one must define the communication scheme used to spread information inside the swarm. Many communication topologies have been proposed to simultaneously accelerate the convergence process and avoid premature convergence to local minima. The most common topologies are: *star*, *ring*, *von Neumann*, *clusters*, *multiple rings*, *Clans*, among others [4].

In the *star* topology, all the particles can share information globally through a fully-connected structure. This topology is often referred as global topology or G_{best} . The *ring* topology is based on a local neighborhood and is also known as L_{best} . In this approach, the particles only share information with their closest neighbors, where the neighborhood is defined based on indexes. The ring topology provides better solutions for multimodal problems when compared to the star topology [4]. Despite of this, the ring topology generally needs more iterations to reach a convergence state.

Some topologies have been proposed to balance the extreme behavior of the G_{best} and L_{best} approaches, such as *Von Neumann* [5], [6] and *clusters* [7]. In the Von Neumann topology, particles are connected in a grid shape, creating a social structure very useful for many optimization problems. Nevertheless, the swarm can still get trapped in local minima in some cases. To avoid this problem, dynamic topologies forming peculiar types of particle grouping, such as *multiple rings* [8] and *clans* [9], [10], have been considered to define the scheme used to exchange information. The Clan Particle Swarm Optimization (ClanPSO) [10] can also be viewed as a cooperative approach.

In addition to that, some approaches have been proposed to dynamically adapt the acceleration constants and the inertia factor during the search process [11], [12]. In line with this is the interesting proposal introduced by Zhan et al. [12], named Adaptive Particle Swarm Optimization (APSO). In which, the state of convergence is estimated at each iteration and the PSO parameters are updated based on this.

Despite the good performance achieved by APSO, the concept was proposed to be applied considering the swarm as a whole. In this paper, we aim to include this ability into a cooperative approach (ClanPSO), where the adaptive

capability is incorporated independently in every sub swarm.

The paper is organized as follow. In section II, we present the basic concepts about Clan Particle Swarm Optimization. In section III, we present the Adaptive Particle Swarm Optimization. In section IV, we introduce our model to include the adaptability into the Clans. In section V, we present the simulation setup. In sections VI and VII, we show a detailed case of study in one multimodal function and some results for other benchmark functions, respectively. Finally, we present our conclusions in section VIII.

II. CLAN PARTICLE SWARM OPTIMIZATION

Clans are groups of individuals united by a kinship. Some clans stipulate a common ancestor to become a kind of symbol, and every individual of the clan will be guided by this symbol. Using this leadership characteristic, a topology was proposed to improve the PSO algorithm performance, the Clan Particle Swarm Optimization (ClanPSO) [9]. This topology is characterized as a set of clans, where each clan is a sub-swarm and uses the global topology internally.

For each iteration, each clan performs a search using a common PSO and marks the particle that had reached the best position of the entire clan. The marked particle is called the leader and this process is called delegation. By using the global information exchange mechanism inside each clan, the delegation process uses the G_{best} information to directly delegate the leader.

After the delegation, the leaders of each clan are put together and another PSO is ran solely with the leaders. This second stage 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, the topology is called ClanPSO-G, whereas the topology is called ClanPSO-L if the L_{best} approach is used for the conference of leaders.

After the conference, the leaders return to their Clans and the new information acquired by the leaders in the conference will be used inside each clan to adjust the velocities of the other particles. One should notice that the leaders do not acquire the best position found by the other leaders so far. Indeed, the leaders solely adjust their positions based on the best position found by the other leaders. It means that a foreign leader does not directly influence another clan. Because of this, a strong attraction to an unique region is avoided and the clan's exploration capacity is preserved.

As the conference of leaders uses only the leaders of each clan to run the new PSO, it will not involve much additional processing since normally the number of Clans is much smaller than the number of particles.

Some efforts were also made to allow the migration of particles among Clans [13]. Despite some improvements achieved for some benchmark functions, there are some problems namely possibility of empty clans.

III. ADAPTIVE PARTICLE SWARM OPTIMIZATION

Zhan *et al.* [12] identified some drawbacks in the PSO behavior concerning the velocity control and the capacity to

escape from local minima. Because of that, they proposed the Adaptive PSO (APSO). The APSO presents a systematic scheme to update the PSO parameters based on fuzzy rules and a learning strategy using elitism. The following steps are executed at each iteration: (i) assessment of the population distribution within the search space; (ii) use this information to estimate what they call the “evolutionary factor”; (iii) classification of the converge stage (or “evolutionary state”) of the population based on the current value of the evolutionary factor and four membership functions; (iv) adaptation of the acceleration parameter in order to speed up the convergence; (v) execution of a learning strategy using elitism to escape from local minima; and finally, (vi) adaptation of the inertia parameter to improve the search capacity. A detailed description of each action is described in the next subsections.

A. Estimating the Evolutionary State

In order to adjust properly the acceleration parameters, one should estimate the evolutionary state of the swarm. This process is divided in two separate steps. In the first, one needs to evaluate the average distance between each particle to all the others by using (3):

$$d_i = \frac{1}{N-1} \sum_{j=1, j \neq i}^N \sqrt{\sum_{k=1}^D (x_i^k - x_j^k)^2}, \quad (3)$$

where N is the number of particles of the swarm and D is the dimensionality of the fitness function. After this, one should evaluate the evolutionary factor (f_{evol}) using (4):

$$f_{evol} = \frac{d_g - d_{min}}{d_{max} - d_{min}} \in [0, 1], \quad (4)$$

where d_g is the average distance between the best particle of the swarm and the other particles, d_{min} is the smallest average distance among all the particles of the swarm and d_{max} is the biggest average distance among all the particles of the swarm. One should notice that the number of particles of the swarm must be higher than three in order to avoid $d_{min} = d_{max}$. The evolutionary factor is bounded by the interval $[0, 1]$. When f_{evol} assumes lower values, it means that all the other particles are close to the best particle of the swarm. On the other hand, higher values means that some particles are not too close to the best particle and they are probably seeking for solutions in other regions of the search space.

B. Classifying the Evolutionary State

Zhan *et al.* [12] proposed to use fuzzy rules to determine the evolutionary state. Figure 1 presents the membership functions used by the APSO. The state with the higher value for the membership function is chosen at each iteration. They defined the following evolutionary states:

1) *Convergence state*: occurs when the value for f_{evol} is too low. It means that all the particles are probably located at the same local minimum. The algorithm should consider to generate diversity.

2) *Exploitation state*: occurs when the value for f_{evol} is low. It means that the average distance between the best particle of the swarm to the other particles is small.

3) *Exploration state*: occurs when the value for f_{evol} is medium. 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 best particle position. In this case, the particles are exploring the search space at large and are not focused to exploit a specific region.

4) *Escape state or Jumping out*: occurs when the value for f_{evol} is high. It means that one or more particles are far from the best particle. It can occur when a particle escapes from a local minimum.

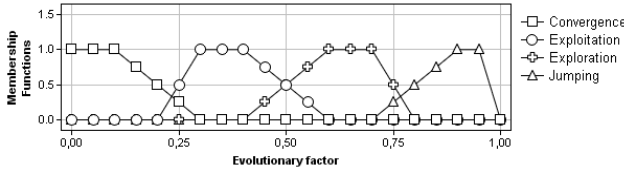


Figure 1. Membership functions used by the APSO.

C. Determining the Acceleration Coefficients

In the APSO, the acceleration coefficients are initialized with values equal to 2.0 ($c_1 = c_2 = 2.0$). c_1 and c_2 are updated along the iterations depending on the current evolutionary state, according table I.

Table I
STRATEGIES TO UPDATE THE ACCELERATION COEFFICIENTS.

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

In the exploration state, the target is to find the maximum number of optimum regions. It can be achieved by increasing the weight of the cognitive term and diminishing the influence of the social term. In the exploitation state, the particles should focus at a particular region. Therefore, the rule to update the acceleration coefficients is similar to the previous, but with a slight behavior. In the convergence state, the social term must be increased to accelerate the convergence toward a single region. However, a premature convergence can occur in some cases when the weight of the cognitive term is decreased. The proposed solution was to slightly increase both acceleration coefficients. In the escape state, as some particle are apart from the current best region, the cognitive term should be diminished in order to allow this particle to slowly return to the current best region.

The step used to update the acceleration coefficients is called acceleration rate (δ). In the APSO proposal [12], δ is obtained by generating a random number using an uniform

probability distribution function in the interval [0.05,0.10]. In the cases when a slight update is necessary, the step is $0.5 \cdot \delta$.

In order to avoid an explosion state, it is necessary to bound the sum $c_1 + c_2$. As the minimum and the maximum value for c_1 and c_2 are $c_{min} = 1.5$ e $c_{max} = 2.5$, when $c_1 + c_2 > c_{min} + c_{max}$, one should update c_1 and c_2 using (5).

$$c_i = \frac{c_i}{c_1 + c_2} \cdot (c_{min} + c_{max}). \quad (5)$$

D. Learning Strategy using Elitism

This strategy aims at maintaining the diversity during the search, by causing a escape state of the best particle when it gets trapped into a local minimum. This stagnation process occurs because the best particle does not have a leader to follow. The solution for is implemented by choosing one of the dimensions of the \vec{x}_{pbest} and applying a gaussian mutation according (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 gaussian mutation is better than the previous one.

$$x_{pbestd}(t+1) = x_{pbestd}(t) + (X_{max}^d - X_{min}^d) \cdot G(\mu, \sigma^2), \quad (6)$$

where (X_{min}^d, X_{max}^d) 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. Empirically, Zhan *et al.* [12] proposed to use $\sigma = 1.0$ in the beginning of the simulations and linearly decrease it to $\sigma = 0.1$ at the end of the simulation.

E. Adaptation of the Inertia Factor

The Inertia Factor ω can be used to adjust the exploration-exploitation abilities of the swarm. In the APSO, ω is obtained as a function of f_{evol} by using (7).

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

IV. AN ADAPTIVE COOPERATIVE APPROACH BASED ON CLAN PARTICLE SWARM OPTIMIZATION

In this paper, we propose to include the adaptation ability, as explained before, to the Clan PSO algorithm. This new algorithm is named *Clan Adaptive Particle Swarm Optimization* (ClanAPSO). The idea is quite simple. We introduced the APSO algorithm concepts in each Clan and we use another independent APSO to perform the conference of leaders. By doing this, we believe that each can adapt itself independently and, as a consequence, the algorithm will avoid to exploit excessively the same spot or get out of a promising region prematurely. The pseudocode of the ClanAPSO is shown in algorithm 1.

V. SIMULATION SETUP

Four benchmark minimization problems were used in the simulations [14] [12]. Rosenbrock and Schwefel 1.2 are unimodal functions, while Rastrigin and Ackley are multimodal functions that contain many local minima. Table II shows the search space, initialization range, and the global minimum for each function.

Algorithm 1: Pseudocode of the algorithm ClanAPSO.

```
1 initialize the swarms;
2 while the stop criterion is not reached do
3   for each clan do
4     execute APSP within the Clan;
5     delegate leader;
6   end
7   create conference of leaders;
8   execute APSP with the leaders;
9 end
```

Table II
USED FUNCTIONS, SEARCH SPACE, INITIALIZATION RANGE, AND OPTIMUM.

Function	Search space	Initialization	Opt.
Rosenbrock	$-30 \leq x_i \leq 30$	$15 \leq x_i \leq 30$	1.0^D
Rastrigin	$-5.12 \leq x_i \leq 5.12$	$2.56 \leq x_i \leq 5.12$	0.0^D
Ackley	$-32 \leq x_i \leq 32$	$16 \leq x_i \leq 32$	0.0^D
Schwefel 1.2	$-100 \leq x_i \leq 100$	$50 \leq x_i \leq 100$	0.0^D

We performed simulations using 10, 30 and 100 dimensions for the Rastrigin function and used 100 dimensions for the other functions (except in section VI.B). All simulations were performed using 30 particles and 300,000 fitness function evaluations. All the results presented in next section were obtained over 30 executions. In all cases, the mean value and standard deviation of the fitness values were evaluated after 30 trials.

VI. ANALYSIS ON ONE MULTIMODAL BENCHMARK FUNCTION

In order to better understand the issues and behaviors generated by the inclusion of the adaptation ability in the multi-swarm approach, we decided to analyze in detail the APSP and ClanAPSP on a single multimodal function. Without loss of generalization, we selected the Rastrigin function.

A. Analysis on the Process of Parameter Adaptation

The first analysis concerns to check the evolution of the acceleration coefficients (c_1 and c_2) and the inertia factor (ω) along the iterations.

Figure 2 presents the APSP-G (using Global topology) behavior. As can be seen, two cognitive epochs occurred. The first one in the very beginning of the simulation and another one starting at the 700th iteration. Despite one can observe some variations on the parameters, these variations are not quite frequent.

Figure 3 presents the behavior of the adaptive parameter of the ClanAPSP-3x10-G running on the Rastrigin function. ClanAPSP-3x10-G means a ClanAPSP with three Clans, where each Clan has 10 particles and the conference of leaders is performed using the global topology.

Analyzing the behavior of the Clans, one can observe a lot of variations in the parameters. Clan 1 and Clan 2 presented an alternate behavior between cognitive and social epochs.

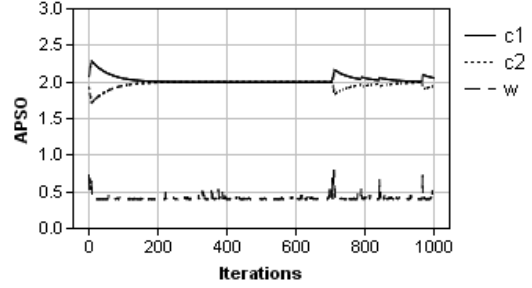


Figure 2. Analysis on c_1 , c_2 and ω parameters adaptation for the APSP algorithm with Global topology running for the Rastrigin function.

Clans 3 presented a behavior mainly focused on the cognitive information.

The conference of leaders presented an opposite behavior when compared to Clan 3. The social term is overweighted during this period. This suggests that the Clans adapt their parameters mostly concerning on the cognitive term, while the leaders exchange information mostly regarding on the social contribution. Probably this observation is related to the necessity to spread information among the Clans.

B. Analysis on the Trace of Solutions

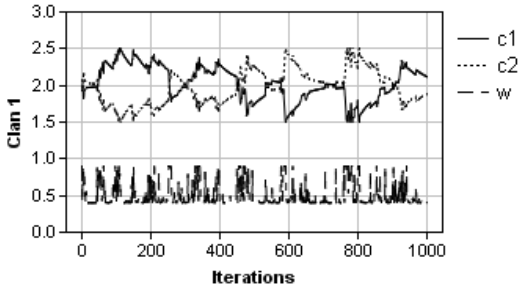
A graphic analysis is performed on the trace of visited points during the search process in order to understand how the algorithms distribute the particles around the search space. We performed this analysis in 2-dimensions to facilitate the visualization. The gray dots are visited positions, while the black dot are the cognitive memories stored along the search process. Figure 4 shows the traces for the Rastrigin function. The traces in the left were generated using global topologies and the traces in the right were generated using local topologies. The algorithms are: (a) PSO, (b) ClanAPSP-3x10 (with three Clans), (c) APSP, (d) ClanAPSP-3x10 (with three Clans).

One can notice that the algorithms with no adaptation ability generate dispersed solutions in the search space, while the adaptive algorithms focused on the best regions. It also can be observed that the ClanAPSP focused on the regions around many minima, but intensified the search around the global optimum (0, 0).

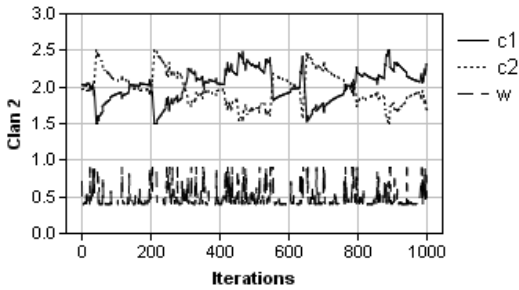
C. Influence of the number of Clans

As the number of Clans plays a role on the performance of the ClanAPSP, one should analyze the behavior of the algorithm ClanAPSP 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, 6 Clans with 5 particles and 10 Clans with 3 particles. For comparison sake, we performed the same simulations for the original ClanAPSP. The algorithms were ran for 1,000 iterations in 100 dimensions.

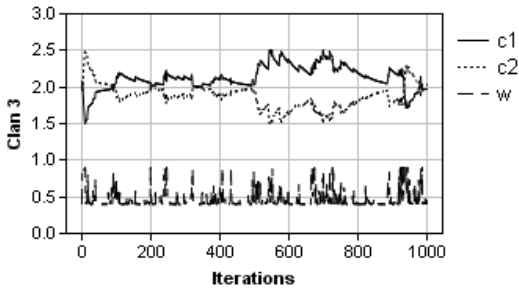
Figure 5 presents the results for the algorithms ClanAPSP and ClanAPSP for the Rastrigin function. One can observe that



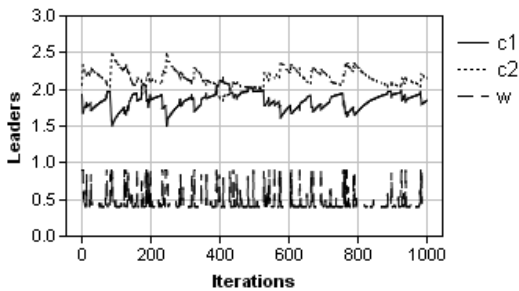
(a)



(b)

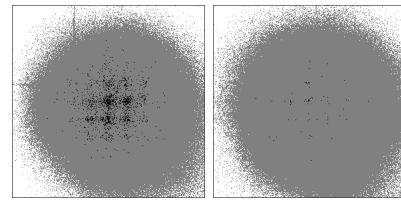


(c)

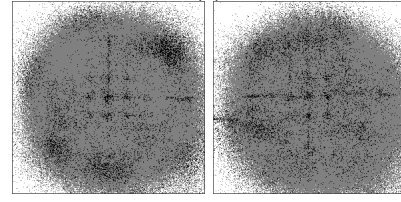


(d)

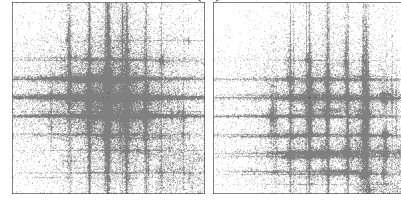
Figure 3. Analysis on the parameter adaptation for the ClanAPSO algorithm with three Clans running for the Rastrigin function for: (a) Clan 1, (b) Clan 2, (c) Clan 3 and (d) Conference of leaders.



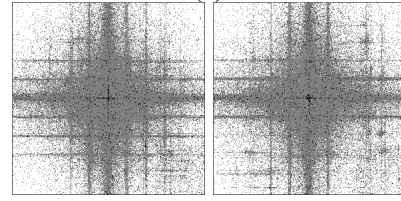
(a)



(b)



(c)



(d)

Figure 4. Scatter plot of generated solutions for the search for different algorithms on the 2-D Rastrigin function. (a) PSO: Global and Local, (b) ClanPSO-3x10: Global and Local, (c) APSO: Global and Local, (d) ClanAPSO-3x10: Global and Local.

all the ClanAPSO configurations outperformed the ClanPSO configurations. Furthermore, one can notice that the number of Clans do not influence a lot on the algorithms performance.

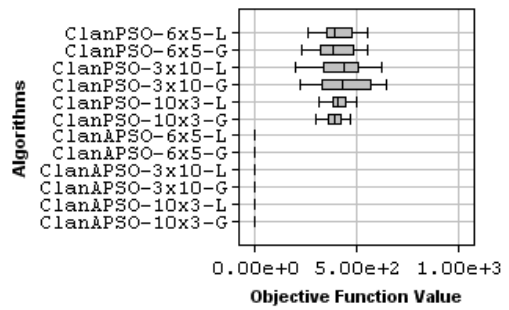


Figure 5. Boxplot of the fitness function for the Rastrigin function varying the number of Clans.

D. Convergence Analysis

We also performed a comparison of various algorithms along iterations to check the convergence speed. We run 10,000 iterations for all the algorithms in the 100 dimensions Rastrigin function.

Figure 6 presents the evolution of Constricted-PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 with global and local topologies. One can notice that the ClanAPSO-6x5, by far, outperforms the other topologies.

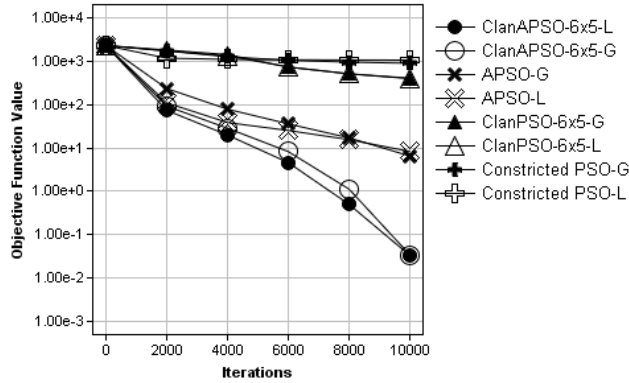


Figure 6. Fitness function evolution of the Constricted-PSO, APSO, ClanPSO-6x5 and ClanAPSO-6x5 for the 100 dimensions Rastrigin function along 10,000 iterations.

VII. RESULTS FOR OTHER BENCHMARK FUNCTION

To assess the performance of ClanAPSO and compare it to other algorithms, we used three other benchmark functions: Schwefel 1.2, Rosenbrock and Ackley. One can note that the best results were achieved by the APSO-L and ClanAPSO-6x5-L approaches for the Schwefel 1.2 and the Rosenbrock functions. On the other hand, the results achieved for the ClanAPSO are better for other multimodal benchmark function, the Ackley function. It suggests that the ClanAPSO returns better results for multimodal functions than the previous approaches.

VIII. CONCLUSIONS

In this paper we included the adaptation ability in a cooperative particle swarm optimization approach. We named our proposal Adaptive Clan Particle Swarm Optimization (ClanAPSO). We showed that different Clans can execute simultaneously in different operation modes which is an interesting feature for convoluted search spaces. We showed that the adaptation ability afforded the algorithm to search more deeply in several local minima, while the cooperative approach intensified the search around the global minimum. When comparing our approach with other previous works, we found it is more effective in multimodal search spaces.

ACKNOWLEDGMENT

The authors thank to POLI-UPE, CAPES and CNPq.

Table III

COMPARISON IN 100-D FOR VARIOUS PSO APPROACHES AND DIFFERENT BENCHMARK FUNCTIONS.

Function	Schwefel 1.2	Rosenbrock	Ackley
APSO-G	2.65e+04 (3.43e+03)	2.28e+02 (9.14e+01)	1.62e-02 (1.27e-02)
APSO-L	3.62e+03 (9.29e+02)	1.42e+02 (7.73e+01)	5.27e-01 (5.75e-01)
ClanAPSO-6x5-G	1.24e+04 (2.52e+03)	1.93e+02 (7.51e+01)	2.78e-04 (1.96e-04)
ClanAPSO-6x5-L	8.96e+03 (1.50e+03)	1.73e+02 (5.87e+01)	1.31e-05 (1.01e-05)
ClanPSO-6x5-G	5.53e+04 (8.39e+03)	1.94e+04 (1.70e+04)	1.32e+01 (7.67e+00)
ClanPSO-6x5-L	5.50e+04 (8.90e+03)	1.29e+04 (1.09e+04)	1.46e+01 (7.60e+00)
Constricted PSO-G	1.88e+07 (5.40e+07)	2.72e+02 (9.67e+01)	2.11e+01 (1.15e-01)
Constricted PSO-L	3.34e+08 (1.83e+08)	2.57e+08 (1.57e+08)	1.98e+01 (1.97e-02)

REFERENCES

- [1] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, 1995, pp. 39–43.
- [2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, 1995.
- [3] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in amultidimensional complex space," *IEEE transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [4] A. Engelbrecht, *Fundamentals of computational swarm intelligence*. John Wiley & Sons, 2006.
- [5] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," *Proceedings of the Evolutionary Computation on*, vol. 1, pp. 1671–1676, 2002.
- [6] E. Peer, F. van den Bergh, and A. Engelbrecht, "Using neighbourhoods with the guaranteed convergence pso," *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, pp. 235–242, 24–26 April 2003.
- [7] R. Mendes, J. Kennedy, and J. Neves, "Watch thy neighbor or how the swarm can learn from its environment," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, 2003, pp. 88–94.
- [8] C. Bastos-Filho, M. Caraciolo, P. Miranda, and D. Carvalho, "Multi-ring Particle Swarm Optimization," in *Proceedings of the 2008 10th Brazilian Symposium on Neural Networks*. IEEE Computer Society Washington, DC, USA, 2008, pp. 111–116.
- [9] D. Carvalho and C. Bastos-Filho, "Clan Particle Swarm Optimization," in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence)*, 2008, pp. 3044–3051.
- [10] D. de Carvalho and C. Bastos-Filho, "Clan particle swarm optimization," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, p. 1, 2009.
- [11] A. Ratnaweera, S. Halgamuge, and H. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.
- [12] 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.
- [13] C. Bastos-Filho, D. Carvalho, E. Figueiredo, and P. de Miranda, "Dynamic Clan Particle Swarm Optimization," in *2009 Ninth International Conference on Intelligent Systems Design and Applications*. IEEE, 2009, pp. 249–254.
- [14] D. Bratton and J. Kennedy, "Defining a Standard for Particle Swarm Optimization," in *Swarm Intelligence Symposium, 2007. IEEE SSCI 2007*, 2007, pp. 120–127.