WCCI 2012 IEEE World Congress on Computational Intelligence
June, 10-15, 2012 - Brisbane, Australia IEEE CEC

Multiple Offspring Sampling in Large Scale Global Optimization

Antonio LaTorre*†, Santiago Muelas*, José-María Peña*
* DATSI, Facultad de Informática, Universidad Politécnica de Madrid
† Instituto Cajal, Centro Superior de Investigaciones Científicas (CSIC)

Abstract—**Continuous optimization is one of the most active research lines in evolutionary and metaheuristic algorithms. Through CEC 2005 to CEC 2011 competitions, many different algorithms have been proposed to solve continuous problems. The advances on this type of problems are of capital importance as many real-world problems from very different domains (biology, engineering, data mining, etc.) can be formulated as the optimization of a continuous function. In this paper we analyze the behavior of a hybrid algorithm combining two heuristics that have been successfully applied to solving continuous optimization problems in the past. We show that the combination of both algorithms obtains competitive results on the proposed benchmark by automatically selecting the most appropriate heuristic for each function and search phase.**

Index Terms—**Continuous Optimization, Hybridization, MOS, MTS, Solis & Wets**

I. Introduction

Continuous optimization is getting more and more attention in the last years. Many real-world problems from different domains (biology, data mining, engineering, etc.) can be formulated as the optimization of a continuous function. These problems have been tackled using Evolutionary Algorithms (EA) [1] or similar metaheuristics [2]. Selecting an appropriate algorithm to solve a continuous optimization problem is not a trivial task. Although a particular algorithm can be configured to perform properly in a given scale of problems (considering the number of variables as their dimensionality), the behavior of the algorithm can degrade as this dimensionality increases, even if the nature of the problem remains the same [3].

In this contribution, the Multiple Offspring Sampling (MOS) framework has been used to combine two different heuristics: the first one of the local searches of the MTS algorithm [2] and the well-known Solis and Wets heuristic [4]. This framework allows the combination of different metaheuristics following an HRH (High-level Relay Hybrid) approach (this nomenclature will be reviewed in Section II) in which the number of evaluations that each algorithm can carry out is dynamically adjusted. The hybrid algorithm will be evaluated on the benchmark problems proposed for the *"Special Session on Evolutionary Computation for Large Scale Global Optimization"* held at the IEEE CEC 2012 [5].

The paper is organized as follows. Section II reviews some relevant work conducted on similar problems. In Section III the algorithm proposed in this work is described in detail, whereas section IV will present the experimental scenario under consideration, describing both the considered benchmark and the execution environment used in our tests. In Section V, the results obtained are presented and discussed, listing the main conclusions derived from this study. Finally, Section VI contains the concluding remarks obtained from this work.

II. Preliminaries

The HRH terminology was introduced in [6], one of the first attempts to define a complete taxonomy of hybrid metaheuristics. This taxonomy is a combination of a hierarchical and a flat classification structured into two levels. The first level defines a hierarchical classification in order to reduce the total number of classes, whereas the second level proposes a flat classification, in which the classes that define an algorithm may be chosen in an arbitrary order. From this taxonomy, the following four basic hybridization strategies can be derived: (a) LRH (Low-level relay hybrid): One metaheuristic is embedded into a single-solution metaheuristic. (b) HRH (High-level relay hybrid): Two metaheuristics are executed in sequence. (c) LTH (Low-level teamwork hybrid): One metaheuristic is embedded into a population-based metaheuristic. (d) HTH (High-level teamwork hybrid): Two metaheuristics are executed in parallel. For this work, we have focused on the HRH group, the one the proposed algorithm belongs to.

In the last years there has been an intense research in HRH and, in particular, in memetic models, combining different types of metaheuristics. However, in this contribution we have followed a slightly different way. Instead of combining a population-based technique with a local search, we have decided to use two different local searches simultaneously. There are two reasons for this approach. First, each of the local searches has obtained remarkable results when combined with other algorithms in [2] and [7], respectively. Second, given the large dimensionality of the proposed benchmark, it seemed a good idea to focus the search on a small number of candidate solutions (ideally one) and use several search strategies to introduce diverse modifications to these solutions.

Finally, in the recent years, there have been several sessions that have focused on large scale global optimization: CEC 2008, ISDA 2009, CEC 2010; and also a special issue in the "Soft Computing - A Fusion of Foundations, Methodologies and Applications" journal. In all these events different local searches have been used in the best performing algorithms [2], [3], [7], [8]. For this reason, an HRH algorithm combining two local searches has been proposed to deal with the benchmark of functions proposed for the *"Special Session on Evolutionary*

Computation for Large Scale Global Optimization" of CEC 2012.

III. Proposed Approach

Multiple Offspring Sampling (MOS) is a framework for the development of Dynamic Hybrid Evolutionary Algorithms [9]. It has been successfully applied to combine different types of algorithms on well-known continuous optimization benchmarks [3], [10].

MOS provides the functional formalization necessary to design this type of algorithms, as well as the tools to identify and select the best performing configuration for the problem under study. In this context, the hybridization of several algorithms can lead to the following two situations:

- A collaborative synergy emerges among the different algorithms that improves the performance of the best one when it is used individually.
- A competitive selection of the best one takes place, in which a similar performance (often the same) is obtained with a minimum overhead.

In MOS, a key term is the concept of *technique*, which is a mechanism, decoupled from the main algorithm, to generate new candidate solutions. This means that, within a MOS-based algorithm, several reproductive mechanisms can be used simultaneously, and it is the main algorithm which selects among the available reproductive techniques the most appropriate for the particular problem and search phase. A more concrete definition for these reproductive mechanisms follows:

Definition A MOS reproductive technique is a mechanism to create new individuals in which: (a) a particular meta-heuristic, (b) an appropriate solution encoding, (c) specific operators (if required), and (d) necessary parameters have been defined.

Furthermore, the use of multiple reproductive mechanisms simultaneously has to be controlled in some way. First, we should define periodical checkpoints in which the participation of each technique in the overall process is adjusted according to its current performance. The periods between these checkpoints are the steps of our algorithm. Then, we should define the number of individuals that each technique j can generate at each step i ($\Pi_i^{(j)}$), which is called its participation ratio. This ratio is uniformly distributed at the beginning of the search process, and it is periodically updated according to a given policy. In the canonical version of MOS, this adjustment is carried out by what is known as a *Participation Function*. These functions can carry out simple static assignments or, more interestingly, dynamic adjustments according to a *Quality Measure* that evaluates how good the offspring of each technique j is during the step i from the point of view of that measure ($Q_i^{(j)}$).

At this point, different measures can be proposed, depending on the concept of quality that we consider. One option would be to consider the Fitness Average of the subset of the best individuals of the offspring generated by each technique as our measure for quality. Other alternatives could be used (Negative Slope Coefficient by [11], Age, Diversity, etc.). If the Fitness Average measure is selected (function *fit* in Equation 1), the quality value computed from the offspring population produced by a technique j during the step i ($O_i^{(j)}$) is obtained as defined in Equation 1.

$$Q_i^{(j)} = \sum_{o \in O_i^{(j)}} \frac{fit(o)}{|O_i^{(j)}|} \tag{1}$$

At the end of each step, the quality of each of the available techniques is recomputed, according to the Quality Function (QF) under consideration. These quality values are then used by a Dynamic Participation Function (PF) to adjust the number of Fitness Evaluations allocated for each technique at each step (Equation 2). This PF computes, at each step, a trade-off factor for each technique, $\Delta_i^{(j)}$, that represents the decrease in participation for the $j-th$ technique at the $i-th$ step, for every technique except the best performing ones. These techniques will increase their participation by the sum of all those $\Delta_i^{(j)}$ divided by the number of techniques with the best quality values.

$$\Pi_i^{(j)} = \begin{cases} \Pi_{i-1}^{(j)} + \eta & \text{if } j \in \Omega, \\ \Pi_{i-1}^{(j)} - \Delta_i^{(j)} & \text{otherwise} \end{cases} \tag{2}$$

$$\eta = \frac{\sum_{k \notin \Omega} \Delta_i^{(k)}}{|\Omega|}$$

$$\Omega = \{l \ / \ Q_i^{(l)} \geq Q_i^{(m)} \ \forall l, m \in [1, n]\}$$

$$Q_i^{max} = max\{Q_i^{(j)} \forall j \in [1, n]\}$$

The above-mentioned $\Delta_i^{(j)}$ values are computed as shown in Equation 3. These $\Delta_i^{(j)}$ factors are computed from the relative difference between the quality of the $best$ and the $j-th$ techniques, n being the number of available techniques. In this equation, ξ represents a reduction factor, i.e., the ratio that is transferred from one technique to the other(s). Finally, a minimum participation ratio can be established to guarantee that all the techniques are represented through all the search. This is done to avoid, if possible, premature convergence to undesired solutions caused by a technique that obtains all the participation in the early steps of the search and quickly converges to poor regions of the solution space, preventing the other techniques to collaborate at later stages of the process, in which they could be more beneficial.

$$\Delta_i^{(j)} = \xi \cdot \frac{Q_i^{max} - Q_i^{(j)}}{Q_i^{max}} \cdot \Pi_{i-1}^{(j)} \quad \forall j \in [1, n] \ / \ j \notin \Omega \tag{3}$$

Finally, the Multiple Offspring Sampling framework allows the development of both HTH (High-level Teamwork Hybrid) and HRH (High-level Relay Hybrid) algorithms (according to Talbi's nomenclature [6]). In the case of the HTH algorithms, two metaheuristics are executed in parallel, working at the same time on the resolution of the problem. On the other hand, in the case of the HRH algorithms, two metaheuristics are

```
1:  Create and evaluate initial solution
2:  Initialize step length: $FEs_0 = \phi$
3:  Uniformly distribute participation among the $n$ used techniques ($T_j$): $\forall j\ \Pi_0^{(j)} = \frac{1}{n}$
    Each technique produces a subset of individuals according to its participation: $FEs_0^{(j)} = \Pi_0^{(j)} \cdot FEs_0$
4:  while number of evaluations not exceeded do
5:      Start new step $i$
6:      Update step length: $FEs_i = FEs_{i-1}$
7:      Update Quality of $T_j$ computed as the average quality of all the individuals created by technique $T_j$ in step $i-1$
8:      Update participation ratios from Quality values computed in Step 7: $\forall j\ \Pi_i^{(j)} = PF(Q_i^{(j)})$
9:      Update $FEs$ allocated for each technique at current step: $\forall j\ FEs_i^{(j)} = \Pi_i^{(j)} \cdot FEs_i$
10:     for every available technique $T_j$ do
11:         while $FEs_i^{(j)}$ not exceeded do
12:             Evolve
13:         end while
14:     end for
15: end while
```

Fig. 1. HRH MOS Algorithm

executed in sequence, one after the other, and changes of the executing algorithm are carried out according to a given policy. As the proposed algorithm is of the HRH type, more attention will be paid to this type of algorithms.

In terms of the MOS framework, the available techniques in a MOS-based HRH hybrid algorithm are used in sequence, one after the other, each of them reusing the output population of the previous technique. This approach fits better when there are non-population-based techniques, such as local searches, or algorithms with very different offspring sampling mechanisms as techniques are not constrained to produce a % of the common population. If different population sizes are used by different techniques, it is the responsibility of the technique to make the population grow/shrink in order to adjust it to its needs and to return a population of an appropriate size to the next technique. For example, if a population-based algorithm is combined with a local search, the latter could select one or more individuals from the output population of the population-based algorithm, modify them as needed and then include them in the original population by means of a predefined elitism procedure.

In this type of algorithms, the search process can be divided into a number of *steps*, which is established at the beginning of the execution [10]. In this approach, each step is assigned a fixed amount of Fitness Evaluations (FEs_i in Algorithm 1) that will not change through the execution of the algorithm. This number of FEs is computed according to a step factor (ϕ in Algorithm 1), that can be static (as in this case) or dynamic. In both cases, the evaluations assigned to each step are distributed by the Participation Function (PF).

Each technique can manage its number of allocated FEs at each step of the algorithm ($FEs_i^{(j)}$) in its own particular way. For example, a population-based technique, such as Differential Evolution, could execute several iterations of the algorithm, whereas a Local Search could decide to spend all its assigned evaluations in improving just one individual. The quality of the new individuals of each technique will be averaged at the end of the whole set of evaluations, as the division of the search into generations depends on each of the techniques.

The pseudocode given in Algorithm 1 summarizes the way an HRH MOS algorithm works. Further information about the MOS framework can be found in [9].

In this contribution, an HRH Dynamic algorithm is proposed. This algorithm combines the first one of the local searches of the MTS algorithm (MTS-LS1) [2] and the well-known Solis and Wets heuristic [4]. These two heuristics have been previously used for large scale global optimization problems, combined with other algorithms. In our experimentation we want to evaluate how well these algorithms can do without the support of a population-based algorithm.

For the adjustment of the participation of each technique in the overall search process, the Participation Function described in Equations 2 and 3 has been used. To measure the quality of the different techniques, the Quality Function in Equation 4 has been proposed. This QF takes into account the Average Fitness Increment of the newly created individuals after a set of allocated Fitness Evaluations.

$$\begin{aligned} Q_i^{(j)} &= \Sigma_{i-1}^{(j)} \\ Q_i^{(j)} &\equiv \text{Quality of technique } T_j \text{ in step } i \\ \Sigma_i^{(j)} &\equiv \text{Average Fitness Increment of } T_j \text{ in step } i \end{aligned} \tag{4}$$

To summarize, the presented algorithm works as follows. All the available techniques are allocated the same number of FEs at the beginning of the execution. At the end of each step, the quality of the new solutions created by each technique is evaluated and, based on this quality, its participation ratio is adjusted accordingly. This participation ratio is used to compute the number of FEs that each technique will be allowed to use in the next step of the search. If a minimum participation ratio has been established, then the number of FEs can not go below this threshold.

IV. Experimental Setup

For the experimentation, the benchmark from the *"Special Session on Evolutionary Computation for Large Scale Global Optimization"* held at the IEEE CEC 2012 has been considered. This benchmark defines 20 continuous optimization functions with different degrees of separability: from completely separable functions to non-separable functions. Detailed information on the benchmark can be found in [5] and at the web page of the organizers of the special session [1].

Table I contains the parameters and the tested values for both the individual techniques and the hybrid algorithm. In

[1]http://staff.ustc.edu.cn/ketang/cec2012/lsgo_competition.htm

particular, we have tested several values for the following parameters, which can dramatically influence on the performance of the algorithm. Regarding the Solis and Wets algorithm, we have focused on the maximum number of successful modifications of the solution (*maxSuccess*) before increasing the adjustment ratio (*delta*), the maximum number of unsuccessful modifications (*maxFailed*) and the increasing and decreasing factors (*adjustSuccess* and *adjustFailed*, respectively). On the other hand, regarding the MTS-LS1 algorithm, we have considered the resetting factors for the SR parameter after one iteration without improvements (*adjustFailed*) or after reaching the minimum value for SR (*adjustMin*) and the adjustment ratios in both directions of the search space (*moveLeft* and *moveRight*, respectively). Finally, regarding the hybrid algorithm, we have studied two parameters: the minimum participation ratio for a technique (*minPart*) and the step factor (*stepFactor*). The values tested in this experimentation have been selected according to previous studies [7], [10] and a preliminary experimentation on this benchmark.

Parameter Values of Solis and Wets	
maxSuccess	**2**, 5, 10
maxFailed	1, 3, **5**
adjustSuccess	**2**, 4, 6
adjustFailed	0.25, 0.5, **0.75**
delta	1.2, **5**, 8
Parameter Values of MTS-LS1	
(initial) SR	50% of the search space
(min) SR	$1e-14$
adjustFailed	2, 3, **5**
adjustMin	**2.5**, 5, 10
moveLeft	0.25, **0.5**, 2
moveRight	**0.25**, 0.5, 2
Parameter Values for the Hybrid Algorithm	
minPart	0%, 1%, **5%**
stepFactor	3000, **18000**, 36000

TABLE I
PARAMETERS VALUES

For the experimentation a fractional design based on orthogonal matrices according to the Taguchi method [12] was chosen in order to conduct a study on the effect of each parameter on the response variable, as well as the effects of interactions between the parameters. This method allows the execution of a limited number of configurations and still reports significant information on the best combination of parameter values. In particular, 27 different configurations were tested and the values for the parameters of the best configuration appear in bold in Table I.

In order to make the results comparable with other algorithms, we have strictly followed the conditions imposed by the benchmark. Therefore, for each combination, 25 independent executions were carried out. The stopping criterion, as defined in the benchmark, was a fixed number of fitness evaluations (3000 times the dimension of the problem, i.e., 3M FEs). The performance criterion (i.e. the response variable) is the distance (error) between the best individual found and the global optimum in terms of fitness value.

Fig. 2. Mean results for F_2

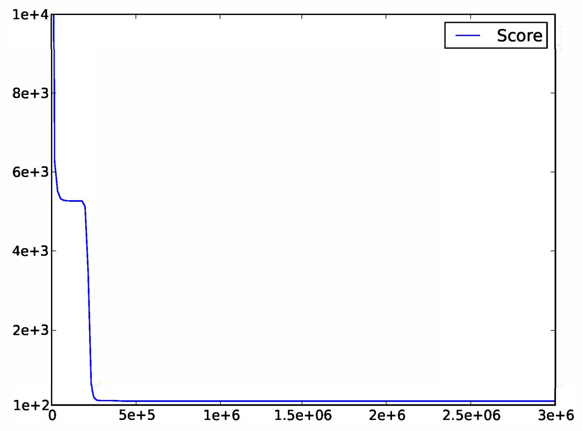
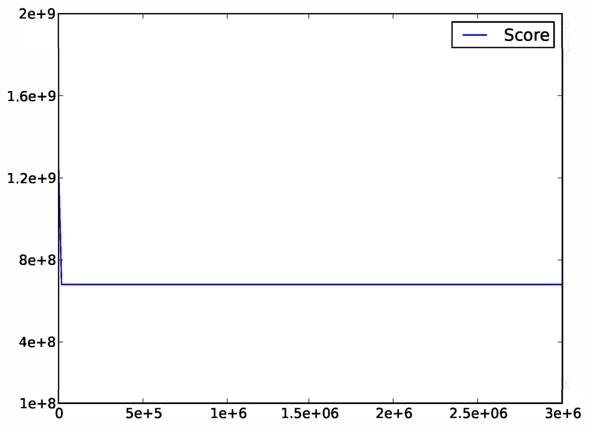


Fig. 3. Mean results for F_5



V. RESULTS AND DISCUSSION

In this section we present and discuss the results of the proposed algorithm on the benchmark used for this special session and competition. This analysis is divided into three parts. First, we show the convergence graphs for several functions in Section V-A. Second, we analyze the behavior of the MOS algorithm on the different groups of functions in Section V-B. Finally, Section V-C presents a comparison of our results with those of the reference algorithms.

A. Convergence Analysis

In this section we provide the convergence graphs of the MOS algorithm on eight selected functions: F_2, F_5, F_8, F_{10}, F_{13}, F_{15}, F_{18} and F_{20}. For each function, a single convergence curve has been plotted using the average results of 25 independent executions. Figures 2-9 show the convergence graphs for functions F_2-F_{20}, respectively. The following characteristics can be observed:

- In functions F_2 and F_{10} there is a quick improvement of the score value at the beginning of the search. Then the

Fig. 4. Mean results for F_8

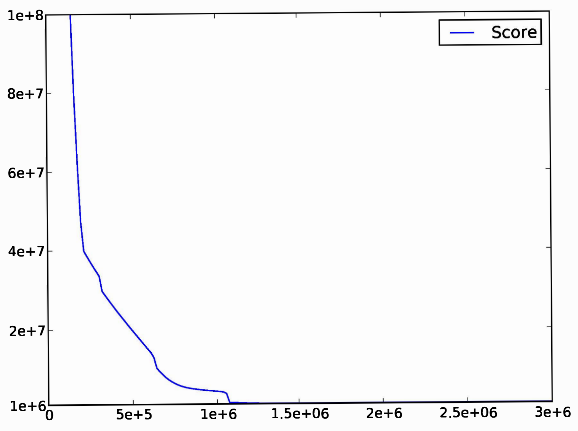


Fig. 5. Mean results for F_{10}

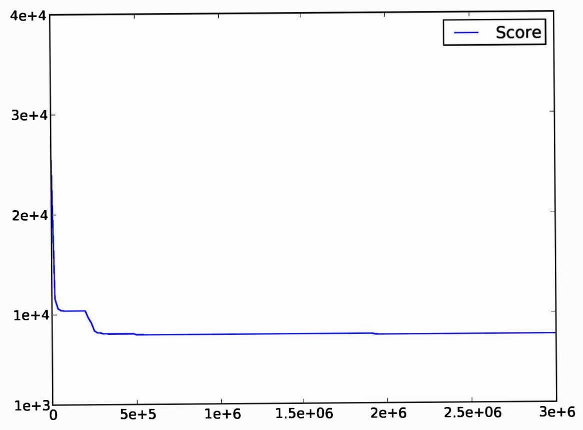


Fig. 6. Mean results for F_{13}

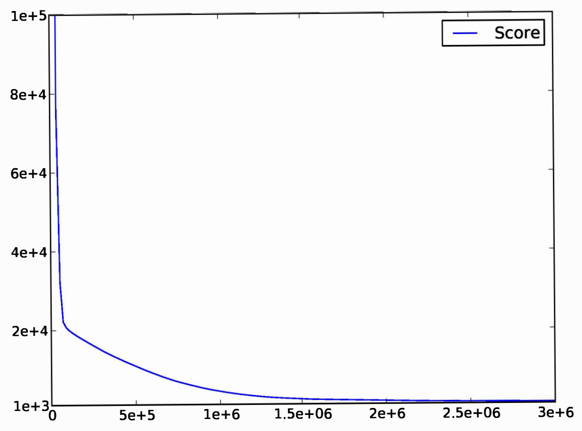


Fig. 7. Mean results for F_{15}

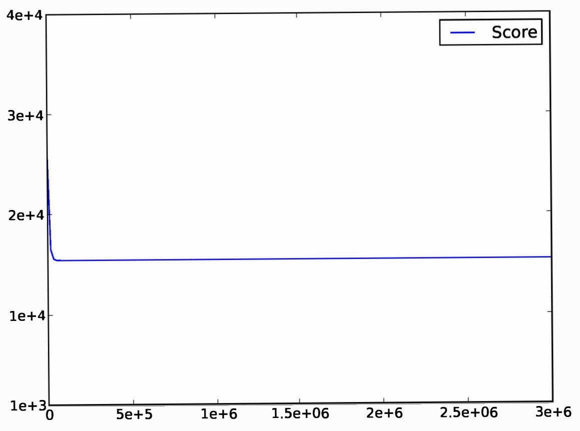


Fig. 8. Mean results for F_{18}

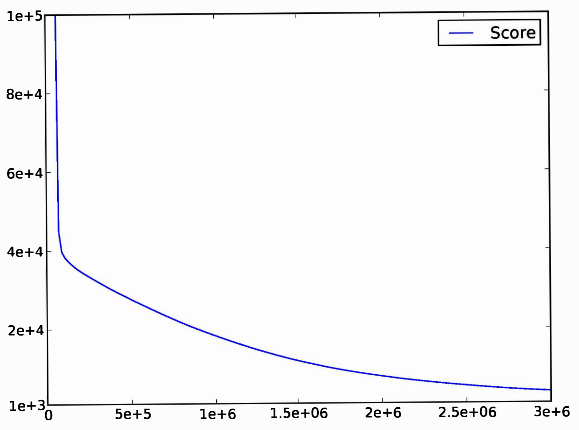
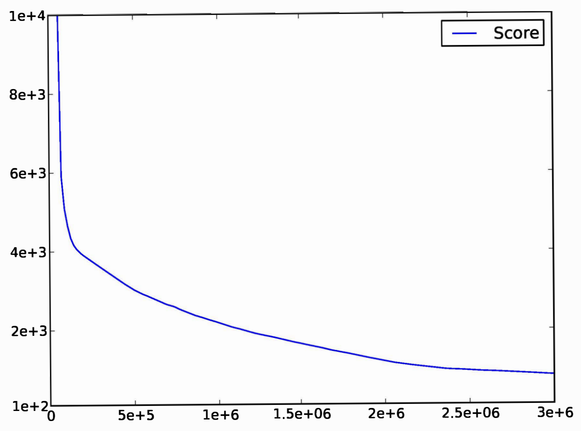


Fig. 9. Mean results for F_{20}



algorithm gets stuck for some time, to introduce a new improvement after several thousands of generations and finally it converges before doing $5e + 05$ evaluations of the allowed $3e + 06$.

- In functions F_5 and F_{15} the algorithm premature converges after a few thousands of fitness evaluations.
- In function F_8 the algorithm continuously improves the score value until approximately $1e + 06$ evaluations. At that point the algorithm converges to a good solution if compared with the reference algorithms.
- In functions F_{13}, F_{18} and F_{20} the algorithm continues to improve the score value through all the search until reaching the maximum number of fitness evaluations.

B. Results of the MOS Algorithm

Table II contains the results of the MOS algorithm on the considered benchmark. From these results we can make the following observations:

- The algorithm is able to solve three functions to the maximum precision (F_1, F_7 and F_{12}) and two other functions to errors close to or below $1e + 00$ (F_3 and F_{17}).
- In most of the functions (except in F_8) the differences between the mean and the median are very low, which implies that our algorithm is rather robust.
- The performance on functions with different number of dependencies (groups 3 and 4 functions as described in [5]) is stable, and the score values do not get significantly worse as the number of dependencies increase.

C. Comparison with Reference Algorithms

Table III compares the results obtained by our algorithm with whose of the reference algorithms: DECC-GG [13] and MLCC [14]. In this table, the best results for both the mean and the median appear in bold to ease its further analysis. In this table we can observe the following:

- The MOS algorithm obtains the best performance in 12 out 20 functions, whereas DECC-GG and MLCC obtain the best results only in 4 and 5 functions, respectively.
- The MOS algorithm seems to perform better on the most complex groups of functions, especially on functions F_7, F_{12} and F_{17}.
- On those functions in which MOS is not the best algorithm it is not very far from the best one, usually no more than one order of magnitude.

VI. Conclusions

In this paper we have proposed a hybrid approach combining the Solis & Wets algorithm and the first of the local searches of the MTS algorithm within the MOS framework. We have conducted an extensive experimental study on a set of 20 large scale continuous functions and compared the results with those of the well-known MLCC and DECC-GG algorithms. This experimentation shows that our proposal is a fast algorithm that obtains very competitive results, especially in the more complex non-separable functions.

Acknowledgments

This work was financed by the Spanish Ministry of Science (TIN2010-21289-C02-02) and supported by the Cajal Blue Brain Project. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the Centro de Supercomputación y Visualización de Madrid (CeSViMa) and the Spanish Supercomputing Network. A. LaTorre gratefully acknowledges the support of the Spanish Ministry of Science and Innovation (MICINN) for its funding throughout the Juan de la Cierva program.

References

[1] T. Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms.* Oxford University Press, Dec. 1995.

[2] L. Y. Tseng and C. Chen, "Multiple Trajectory Search for Large Scale Global Optimization," in *Proceedings of the 10th IEEE Congress on Evolutionary Computation, CEC 2008 (IEEE World Congress on Computational Intelligence).* IEEE Press, Jun. 2008, pp. 3052–3059.

[3] A. LaTorre, S. Muelas, and J. M. Peña, "A MOS-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test," *Soft Computing-A Fusion of Foundations*, vol. 15, no. 11, pp. 2187–2199, 2011.

[4] F. J. Solis and R. J. B. Wets, "Minimization by Random Search Techniques," *Mathematics of Operations Research*, vol. 6, no. 1, pp. 19–30, Feb. 1981.

[5] K. Tang, X. Li, P. Suganthan, Z. Yang, and T. Weise, "Benchmark Functions for the CEC'2010 Special Session and Competition on Large Scale Global Optimization," Tech. Rep., 2010.

[6] E.-G. Talbi, "A Taxonomy of Hybrid Metaheuristics," *Journal of Heuristics*, vol. 8, no. 5, pp. 541–564, Sep. 2002.

[7] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains: Memetic Algorithm Based on Local Search Chains for Large Scale Continuous Global Optimization," *Proceedings of the 2010 IEEE Congress on Evolutionary Computation, CEC 2010*, pp. 1–8, 2010.

[8] S. Muelas, A. LaTorre, and J. M. Peña, "A Memetic Differential Evolution Algorithm for Continuous Optimization," in *Proceedings of the 9th International Conference on Intelligent Systems Design and Applications, ISDA 2009.* IEEE Press, Nov. 2009, pp. 1080–1084.

[9] A. LaTorre, "A Framework for Hybrid Dynamic Evolutionary Algorithms: Multiple Offspring Sampling (MOS)," Ph.D. dissertation, Universidad Politécnica de Madrid, Nov. 2009.

[10] A. LaTorre, S. Muelas, and J. M. Peña, "Benchmarking a MOS-based algorithm on the BBOB-2010 Noiseless Function Testbed," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation Conference.* ACM, 2010, pp. 1649–1656.

[11] L. Vanneschi, M. Tomassini, P. Collard, and S. Vérel, "Negative Slope Coefficient : A Measure to Characterize Genetic Programming Fitness Landscapes," in *Proceedings of the 9th European conference on Genetic Programming, EuroGP 2006*, Apr. 2006, pp. 179–189.

[12] G. Taguchi, S. Chowdhury, and Y. Wu, *Taguchi's Quality Engineering Handbook.* John Wiley.

[13] Z. Yang, K. Tang, and X. Yao, "Large Scale Evolutionary Optimization Using Cooperative Coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[14] Z. Yang, K. Tang, and X. Yao, "Multilevel Cooperative Coevolution for Large Scale Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation 2008, CEC 2008*, 2008, pp. 1663–1670.

TABLE II

Experimental results with MOS

		F_1	F_2	F_3	F_4	F_5	F_6	F_7
	Best	4.94e-01	4.81e+03	1.42e-03	5.62e+11	3.60e+08	1.98e+07	2.23e-01
	Median	1.28e+02	5.20e+03	1.29e+00	1.21e+12	6.86e+08	1.99e+07	2.63e-01
FEs = 1.2e+05	Worst	6.23e+02	5.64e+03	2.50e+00	2.01e+12	1.01e+09	2.00e+07	3.05e+00
	Mean	1.53e+02	5.26e+03	1.34e+00	1.19e+12	6.81e+08	1.99e+07	5.46e-01
	Std	1.55e+02	2.20e+02	8.40e-01	3.54e+11	1.42e+08	5.67e+04	6.37e-01
	Best	0.00e+00	1.69e+02	2.17e-12	7.71e+10	3.60e+08	1.98e+07	0.00e+00
	Median	0.00e+00	1.95e+02	1.29e+00	1.46e+11	6.86e+08	1.99e+07	0.00e+00
FEs = 6.0e+05	Worst	0.00e+00	2.41e+02	2.50e+00	2.98e+11	1.01e+09	2.00e+07	0.00e+00
	Mean	0.00e+00	1.97e+02	1.12e+00	1.54e+11	6.81e+08	1.99e+07	0.00e+00
	Std	0.00e+00	1.59e+01	1.00e+00	5.21e+10	1.42e+08	5.67e+04	0.00e+00
	Best	0.00e+00	1.69e+02	2.17e-12	7.36e+09	3.60e+08	1.98e+07	0.00e+00
	Median	0.00e+00	1.95e+02	1.29e+00	1.88e+10	6.86e+08	1.99e+07	0.00e+00
FEs = 3.0e+06	Worst	0.00e+00	2.41e+02	2.50e+00	3.99e+10	1.01e+09	2.00e+07	0.00e+00
	Mean	0.00e+00	1.97e+02	1.12e+00	1.91e+10	6.81e+08	1.99e+07	0.00e+00
	Std	0.00e+00	1.59e+01	1.00e+00	8.08e+09	1.42e+08	5.67e+04	0.00e+00
		F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}
	Best	3.39e+07	1.90e+08	9.92e+03	1.98e+02	8.96e+04	4.35e+03	4.55e+08
	Median	3.94e+07	2.43e+08	1.03e+04	1.99e+02	1.03e+05	1.92e+04	5.46e+08
FEs = 1.2e+05	Worst	4.50e+08	3.36e+08	1.10e+04	2.01e+02	1.39e+05	5.24e+04	6.95e+08
	Mean	1.28e+08	2.47e+08	1.04e+04	1.99e+02	1.05e+05	1.94e+04	5.43e+08
	Std	1.39e+08	3.79e+07	2.52e+02	4.52e-01	1.08e+04	1.12e+04	6.28e+07
	Best	8.61e+03	3.27e+07	7.51e+03	1.98e+02	7.49e+01	1.25e+03	8.17e+07
	Median	8.56e+06	4.52e+07	7.83e+03	1.99e+02	9.63e+01	7.84e+03	1.03e+08
FEs = 6.0e+05	Worst	1.02e+08	6.25e+07	1.03e+04	2.01e+02	1.29e+02	3.13e+04	1.28e+08
	Mean	1.42e+07	4.52e+07	7.96e+03	1.99e+02	9.89e+01	8.95e+03	1.01e+08
	Std	2.25e+07	7.49e+06	5.29e+02	4.52e-01	1.42e+01	7.73e+03	1.06e+07
	Best	9.97e-03	7.16e+06	7.51e+03	1.98e+02	0.00e+00	2.20e+02	1.60e+07
	Median	2.74e-01	8.83e+06	7.83e+03	1.99e+02	0.00e+00	1.18e+03	1.85e+07
FEs = 3.0e+06	Worst	3.99e+06	1.02e+07	8.35e+03	2.01e+02	0.00e+00	3.21e+03	1.99e+07
	Mean	1.12e+06	8.78e+06	7.86e+03	1.99e+02	0.00e+00	1.36e+03	1.82e+07
	Std	1.79e+06	1.01e+06	2.43e+02	4.52e-01	0.00e+00	9.37e+02	1.18e+06
		F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}	
	Best	1.43e+04	3.97e+02	4.27e+05	2.33e+04	2.85e+06	2.14e+03	
	Median	1.54e+04	3.97e+02	5.32e+05	3.83e+04	3.24e+06	2.88e+03	
FEs = 1.2e+05	Worst	1.64e+04	3.98e+02	6.20e+05	5.00e+04	3.73e+06	1.48e+04	
	Mean	1.54e+04	3.97e+02	5.33e+05	3.75e+04	3.27e+06	4.35e+03	
	Std	5.35e+02	2.10e-01	4.24e+04	6.94e+03	1.83e+05	3.59e+03	
	Best	1.43e+04	3.97e+02	6.57e+03	1.36e+04	7.12e+05	1.28e+03	
	Median	1.54e+04	3.97e+02	9.36e+03	2.51e+04	8.09e+05	1.81e+03	
FEs = 6.0e+05	Worst	1.64e+04	3.98e+02	1.11e+04	3.53e+04	9.46e+05	1.05e+04	
	Mean	1.54e+04	3.97e+02	9.41e+03	2.53e+04	8.17e+05	2.81e+03	
	Std	5.36e+02	2.10e-01	8.61e+02	6.41e+03	4.68e+04	2.74e+03	
	Best	1.43e+04	3.97e+02	2.67e-05	1.09e+03	2.81e+04	3.28e+02	
	Median	1.54e+04	3.97e+02	4.83e-05	3.55e+03	3.40e+04	7.26e+02	
FEs = 3.0e+06	Worst	1.64e+04	3.98e+02	5.73e-05	1.06e+04	3.99e+04	1.86e+03	
	Mean	1.54e+04	3.97e+02	4.66e-05	3.91e+03	3.41e+04	8.31e+02	
	Std	5.36e+02	2.10e-01	6.24e-06	2.18e+03	2.63e+03	3.76e+02	

TABLE III
Comparison with Reference Algorithms, FEs = 3.0e6

		F_1	F_2	F_3	F_4	F_5	F_6	F_7
	Best	1.63e-07	1.25e+03	1.20e+00	7.78e+12	1.50e+08	3.89e+06	4.26e+07
	Median	2.86e-07	1.31e+03	1.39e+00	1.51e+13	**2.38e+08**	**4.80e+06**	1.07e+08
DECC-CG	Worst	4.84e-07	1.40e+03	1.68e+00	2.65e+13	4.12e+08	7.73e+06	6.23e+08
	Mean	2.93e-07	1.31e+03	1.39e+00	1.70e+13	**2.63e+08**	**4.96e+06**	1.63e+08
	Std	8.62e-08	3.26e+01	9.73e-02	5.37e+12	8.44e+07	8.02e+05	1.37e+08
	Best	0.00e+00	1.73e-11	1.28e-13	4.27e+12	2.15e+08	5.85e+06	4.16e+04
	Median	**0.00e+00**	**6.43e-11**	**1.46e-13**	1.03e+13	3.92e+08	1.95e+07	5.15e+05
MLCC	Worst	3.83e-26	1.09e+01	1.86e-11	1.62e+13	4.87e+08	1.98e+07	2.78e+06
	Mean	**1.53e-27**	**5.57e-01**	**9.88e-13**	9.61e+12	3.84e+08	1.62e+07	6.89e+05
	Std	7.66e-27	2.21e+00	3.70e-12	3.43e+12	6.93e+07	4.97e+06	7.37e+05
	Best	0.00e+00	1.69e+02	2.17e-12	7.36e+09	3.60e+08	1.98e+07	0.00e+00
	Median	**0.00e+00**	1.95e+02	1.29e+00	**1.88e+10**	6.86e+08	1.99e+07	**0.00e+00**
MOS	Worst	0.00e+00	2.41e+02	2.50e+00	3.99e+10	1.01e+09	2.00e+07	0.00e+00
	Mean	**0.00e+00**	1.97e+02	1.12e+00	**1.91e+10**	6.81e+08	1.99e+07	**0.00e+00**
	Std	0.00e+00	1.59e+01	1.00e+00	8.08e+09	1.42e+08	5.67e+04	0.00e+00
		F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}
	Best	6.37e+06	2.66e+08	1.03e+04	2.06e+01	7.78e+04	1.78e+03	6.96e+08
	Median	6.70e+07	3.18e+08	1.07e+04	**2.33e+01**	8.87e+04	3.00e+03	8.07e+08
DECC-CG	Worst	9.22e+07	3.87e+08	1.17e+04	2.79e+01	1.07e+05	1.66e+04	9.06e+08
	Mean	6.44e+07	3.21e+08	1.06e+04	**2.34e+01**	8.93e+04	5.12e+03	8.08e+08
	Std	2.89e+07	3.38e+07	2.95e+02	1.78e+00	6.87e+03	3.95e+03	6.07e+07
	Best	4.51e+04	8.96e+07	2.52e+03	1.96e+02	2.42e+04	1.01e+03	2.62e+08
	Median	4.67e+07	1.24e+08	**3.16e+03**	1.98e+02	3.47e+04	1.91e+03	3.16e+08
MLCC	Worst	9.06e+07	1.46e+08	5.90e+03	1.98e+02	4.25e+04	3.47e+03	3.77e+08
	Mean	4.38e+07	1.23e+08	**3.43e+03**	1.98e+02	3.49e+04	2.08e+03	3.16e+08
	Std	3.45e+07	1.33e+07	8.72e+02	6.98e-01	4.92e+03	7.27e+02	2.77e+07
	Best	9.97e-03	7.16e+06	7.51e+03	1.98e+02	0.00e+00	2.20e+02	1.60e+07
	Median	**2.74e-01**	**8.83e+06**	7.83e+03	1.99e+02	**0.00e+00**	**1.18e+03**	**1.85e+07**
MOS	Worst	3.99e+06	1.02e+07	8.35e+03	2.01e+02	0.00e+00	3.21e+03	1.99e+07
	Mean	**1.12e+06**	**8.78e+06**	7.86e+03	1.99e+02	**0.00e+00**	**1.36e+03**	**1.82e+07**
	Std	1.79e+06	1.01e+06	2.43e+02	4.52e-01	0.00e+00	9.37e+02	1.18e+06
		F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}	
	Best	1.09e+04	5.97e+01	2.50e+05	5.61e+03	1.02e+06	3.59e+03	
	Median	1.18e+04	**7.51e+01**	2.89e+05	2.30e+04	1.11e+06	3.98e+03	
DECC-CG	Worst	1.39e+04	9.24e+01	3.26e+05	4.71e+04	1.20e+06	5.32e+03	
	Mean	1.22e+04	**7.66e+01**	2.87e+05	2.46e+04	1.11e+06	4.06e+03	
	Std	8.97e+02	8.14e+00	1.98e+04	1.05e+04	5.15e+04	3.66e+02	
	Best	5.30e+03	2.08e+02	1.38e+05	2.51e+03	1.21e+06	1.70e+03	
	Median	**6.89e+03**	3.95e+02	1.59e+05	4.17e+03	1.36e+06	2.04e+03	
MLCC	Worst	1.04e+04	3.97e+02	1.86e+05	1.62e+04	1.54e+06	2.34e+03	
	Mean	**7.11e+03**	3.76e+02	1.59e+05	7.09e+03	1.36e+06	2.05e+03	
	Std	1.34e+03	4.71e+01	1.43e+04	4.77e+03	7.35e+04	1.80e+02	
	Best	1.43e+04	3.97e+02	2.67e-05	1.09e+03	2.81e+04	3.28e+02	
	Median	1.54e+04	3.97e+02	**4.83e-05**	**3.55e+03**	**3.40e+04**	**7.26e+02**	
MOS	Worst	1.64e+04	3.98e+02	5.73e-05	1.06e+04	3.99e+04	1.86e+03	
	Mean	1.54e+04	3.97e+02	**4.66e-05**	**3.91e+03**	**3.41e+04**	**8.31e+02**	
	Std	5.36e+02	2.10e-01	6.24e-06	2.18e+03	2.63e+03	3.76e+02	