A Hybrid Metaheuristic Based on Differential Evolution and Local Search with Quadratic Interpolation

María Laura Tardivo[1], Leticia Cagnina[2], Guillermo Leguizamón[2]

[1] Department of Computer Science, Universidad Nacional de Río Cuarto, Río Cuarto, Córdoba, Argentina
lauratardivo@dc.exa.unrc.edu.ar
[2] Department of Computer Science, Universidad Nacional de San Luis, San Luis, Argentina
lcagnina@unsl.edu.ar
legui@unsl.edu.ar

Abstract. The use of Local Search technique in combination with other methods is often an effective way for increasing the efficiency of a global optimization algorithm. In this paper we present an hybrid version that integrates Differential Evolution with Local Search, applying the *Quadratic Interpolation* formula for determining the neighborhood in which to explore towards better solutions.
We present *DE+LS(1)* in which the closer neighborhood to the best population individual is explored, and *DE+LS(2)* in which the neighborhood of the two best population individuals is examined. The results showed that with *DE+LS(2)* improvements are not significant, but using *DE+LS(1)* an improvement is achieved, especially for large dimensions, in terms of solutions quality and speed of convergence.

Keywords: Differential Evolution, Local Search, Quadratic Interpolation, Optimization.

1 Introduction

Many optimization problems of real life are complex, expensive to solve and sometimes become untreatable. In general, they can not be solved exactly in a reasonable amount of time. Alternatively it is possible to use algorithms to approximate the results to the exact solutions.

According to [10], metaheuristics represent a family of approximated optimization techniques that provide good solutions to complex problems computed in a reasonable time. In [3] they are defined as solutions methods that implement an interaction between local improvement procedures and high level strategies to create a process able to escape from local optimum and search in the good solutions area. Osman and Kelly [8] consider them as a set of concepts

that can be used to define heuristic methods applicable to a wide variety of problems.

Metaheuristics family is divided into two categories formed according to the number of solutions used: single solution based metaheuristics (also known as trajectory metaheuristics) and population based metaheuristics. The first one starts with a single solution that is perturbed to explore the search space of the problem addressed. The latter is characterized by working with a set of solutions, called population, represented by individuals who interact with each other to make the search process. A simple and well-known single solution method is Local Search, which is generally used in combination with other techniques. Within population metaheuristics Differential Evolution can be found as a method of common choice.

This paper presents a hybrid technique that combines two algorithms applied to large scale optimization problems. We propose to use Local Search with Quadratic Interpolation to improve the performance of a classical version of a Differential Evolution algorithm.

The goal of this study is to demonstrate how Local Search scheme can increase the performance of the Differential Evolution algorithm.

The rest of this paper is organized as follows: Sections 2 and 3 describe and characterize the methods used to construct the proposed algorithm. Section 4 the proposal is specified, stating all the components that interact in their execution. Subsequently, the results analysis is presented, including non parametric statistic tests. Finally, a conclusion of all the work done and future works is presented.

2 Local Search

Within single solution based methods, Local Search (LS) is possibly the oldest and the most operationally simple [10]. Perhaps, this is the reason why it is widely used in practice to solve optimization problems.

Local Search starts at a given initial solution. At each iteration, the current solution is replaced by a neighbor that improves the objective function. The search stops when all candidate neighbors are worse than the current solution, meaning that a local optimum is reached. A basic scheme of Local Search is illustrated in Algorithm 1.

Algorithm 1: Template of a Local Search algorithm.

$s = s_0$; /* Generate initial solution s_0 */
While not Termination_Criterion **Do**
  Generate ($N(s)$) ; /* Generation of candidate neighbors*/
  If there is no better neighbor **Then** Stop ;
  $s = s'$; /* Select a better neighbor $s' \in N(s)$ */
Endwhile
Output: Final solution found (local optima).

Although LS is simple and easy to code, one of the main disadvantages is that it does not obtain good performance when the objective function has many

local optima. If the function is highly multimodal, as in the case of the most of the optimization problems, it is not an effective method to use. For this reason, many applications use it in combination with other metaheuristics, in order to exploit Local Search potentialities applied not only to single solution based metaheuristics ([5], [4]), but also to population based metaheuristics such as Particle Swarm Optimization ([15]) or Differential Evolution ([1]).

3 Differential Evolution

The Differential Evolution (DE) algorithm was proposed by Rainer Storn and Kenneth Price in 1995 [9]. It is a population based optimizer that starts generating a population of NP D-dimensional vectors whose initial values are randomly obtained based on the limits defined by the inputs of the algorithm.

Each of the NP individuals belongs to a generation g, i.e., let $X_{i,g} = (x^1_{i,g}, ..., x^D_{i,g})$ a population individual, with $i = 1, ...,$NP where the index i denotes *i-th* population individual and g determines the generation to which the individual belongs.

The main idea of the method is to use difference vectors in order to modify the population vector. This idea has been integrated into a recombination operator of two or more solutions, and a self-referential mutation operator to guide the search towards "good" solutions.

The main DE operators are explained below:

Mutation: After initialization, DE mutates and recombines the current population to produce another one of NP individuals. The mutation process begins in each generation selecting three random individuals $X_{r_1,g}, X_{r_2,g}$, and $X_{r_3,g}$. The *i-th* individual is perturbed using the following strategy:

$$\text{“}DE/rand/1\text{”} : V_{i,g+1} = X_{r3,g} + F * (X_{r1,g} – X_{r2,g}), \tag{1}$$

where $i = 1, ...,$NP and $r_1 \neq r_2 \neq r_3 \neq i$. The constant F represents a scaling factor and controls the difference amplification between individuals r_1 and r_2, and it is used to avoid stagnation in the search process. The formula (1) is the most general of the mutation strategies. Additionally, the original DE version proposes four strategies:

“DE/best/1”: $V_{i,g+1} = X_{best,g} + F * (X_{r_1,g} – X_{r_2,g})$

“DE/rand-to-best/1”: $V_{i,g+1} = X_{r_1,g} + F * (X_{best,g} – X_{r_2,g}) + F * (X_{r_3,g} – X_{r_4,g})$

“DE/best/2”: $V_{i,g+1} = X_{best,g} + F * (X_{r_1,g} – X_{r_2,g}) + F * (X_{r_3,g} – X_{r_4,g})$

“DE/rand/2”: $V_{i,g+1} = X_{r_1,g} + F * (X_{r_2,g} – X_{r_3,g}) + F * (X_{r_4,g} – X_{r_5,g})$

The indexes r_1, r_2, r_3, r_4, r_5 are integers numbers different from each other, randomly generated in the range [1, NP]. $X_{best,g}$ is the best individual, i.e., it has the best value of the objective function evaluation among all individuals of current generation g.

***Crossover*:** After the mutation phase, the perturbed individual $V_{i,g+1} = (v^1_{i,g+1}, ..., v^D_{i,g+1})$ and the individual $X_{i,g} = (x^1_{i,g}, ..., x^D_{i,g})$ are involved in

the crossover operation, generating a new vector $U_{i,g+1} = (u^1_{i,g+1}, ..., u^D_{i,g+1})$, denominated "trial vector", and obtained in the following way:

$$U^j_{i,g+1} = \begin{cases} v^j_{i,g+1} & \text{if } rand_j \leq Cr \text{ or } j = k \\ x^j_{i,g} & \text{in other case,} \end{cases}$$

where $j = 1, ..., D$, and $k \in \{1, ..., D\}$. The latter is a randomly generated index chosen for each individual. The constant Cr, denominated *crossover factor*, is a parameter of the algorithm defined by the user. Cr belongs to the range $[0, 1]$ and is used to control the values fraction that are copied from the mutant vector V. $rand_j$ is the output of a uniformly distributed random number generator. If this random number is less or equal to Cr the trial vector component is inherit from the mutated vector $V_{i,g+1}$; if not, this value is copied from the vector $X_{i,g}$.

It can be seen that the value k is taken from the mutated vector to ensure that the trial vector is not exactly equal to its source vector $X_{i,g}$.

Each method mentioned before is associated to a specific crossing type, according to the mechanism by which the mutation is performed. These are binomial or exponential crossover operators. The former integrates only some of the vector components, while in the second all the individual components are integrated.

***Selection*:** This phase determines which element will be part of the next generation. The objective function of each trial vector $U_{i,g+1}$ is evaluated and compared with the value of the objective function for its counterpart $X_{i,g}$ in the current population. If the trial vector has less or equal objective function target value (for minimization problems) it will replace the vector $X_{i,g}$ in the population of next generation. The scheme followed is:

$$X_{i,g+1} = \begin{cases} U_{i,g+1} & \text{if } f(U_{i,g+1}) \leq f(X_{i,g}) \\ X_{i,g} & \text{in other case.} \end{cases}$$

The three stages mentioned above are repeated from generation to generation until the specified termination criterion is satisfied.

Due to the potentialities provided by the DE utilization for optimization problems, in recent years it have been proposed numerous variations and methods that attempt to improve the performance of the classic technique. Among them are those trying to adapt DE parameters such as self-adjusting ([12],[16], [14], [2]); others using different mechanisms to optimize the individuals selection for the mutation and selection phases ([6]), and some combining both methods ([13]).

It has also been investigated the utilization of Differential Evolution in conjuction with other methods, conforming hybrid algorithms. One of them combines DE with single solution based metaheuristics ([1] [7]).

4 The proposed algorithm

The proposed algorithm is based on [1], in which two Local Search strategies are used for the neighborhood generation. One of them, named Interpolated Local

Search, has its foundations in the Quadratic Interpolation mathematical formula. It consists of obtaining the parabolic curve determined by three different, randomly selected points in the plane. The point located at the minimum of that curve is evaluated in the following way:

$$Y = \frac{1}{2}\frac{(X_{r_1}^2 - X_{r_2}^2)f(X_{r_3}) + (X_{r_2}^2 - X_{r_3}^2)f(X_{r_1}) + (X_{r_3}^2 - X_{r_1}^2)f(X_{r_2})}{(X_{r_1} - X_{r_2})f(X_{r_3}) + (X_{r_2} - X_{r_3})f(X_{r_1}) + (X_{r_3} - X_{r_1})f(X_{r_2})}, \quad (2)$$

where X_{r_1}, X_{r_2} and X_{r_3} are different each other; and f is the function to approximate.

Then, it was decided to apply the formula component by component as follows: Let $Y_g = (y_g^1, .., y_g^D)$, $X_{r_1,g} = (x_{r_1,g}^1, .., x_{r_1,g}^D)$, $X_{r_2,g} = (x_{r_2,g}^1, .., x_{r_2,g}^D)$ and $X_{r_3,g} = (x_{r_3,g}^1, .., x_{r_3,g}^D)$. Each component y_g^i is calculated using the following formula:

$$y_g^i = \frac{1}{2}\frac{((x_{r_1,g}^i)^2 - (x_{r_2,g}^i)^2)f(X_{r_3,g}) + ((x_{r_2,g}^i)^2 - (x_{r_3,g}^i)^2)f(X_{r_1,g}) + ((x_{r_3,g}^i)^2 - (x_{r_1,g}^i)^2)f(X_{r_2,g})}{(x_{r_1,g}^i - x_{r_2,g}^i)f(X_{r_3,g}) + (x_{r_2,g}^i - x_{r_3,g}^i)f(X_{r_1,g}) + (x_{r_3,g}^i - x_{r_1,g}^i)f(X_{r_2,g})} \quad (3)$$

It can be observed that the main idea is based on trying to approximate each of the vector dimensions. The selection of the individuals $X_{r_1,g}$, $X_{r_2,g}$ and $X_{r_3,g}$ was done with the goal of guiding the search close to the neighborhood of the best population individual.

The proposal developed applies two selection schemes in Local Search. The formula (3) is applied with the following individuals:

- *DE+LS(1)*: Randomly selecting two individuals, in combination with the best individual found.
- *DE+LS(2)*: Randomly selecting one individual, in combination with the two best population individuals found.

The developed process applies the classic DE scheme until the selection phase. After each generation completion (including initial generation) Local Search is applied. The scheme is summarized in the following steps:

1. Generate the initial population.
2. Compute the objective function values for the individuals.
3. **while** not Finalization_Criterion **do**
4. Apply LS with the corresponding scheme (*DE+LS(1)* or *DE+LS(2)*).
5. **if** the new generated individual is best than the current best **then**
6. replace the best (and update the second best, if using *DE+LS(2)*).
7. **else**
8. continue like the classic DE, i.e.:
9. _ Apply Mutation, Crossover and Selection.
10. _ Update population for next generation.
11. **end**
12. **end**

Local Search uses the first improvement strategy for the selection of a new neighbor. In DE the finalization condition is determined by the maximum number of iterations considered, which is a parameter of the algorithm. For all tests 6000 iterations were considered.

The determination of the constant factors F and Cr was made based on earlier studies on ED.[3] Then, $F = 0.5$ and $Cr = 0.3$ were considered.

5 Results analysis

The performance for the proposed algorithm was tested with a set of scalable functions, obtained from [11]. For each of them, 30 executions were carried out with different seeds. The sizes of the problems considered were 30, 100, 500, with a population made up with 100 individuals (NP=100). Table 1 shows a brief description of each of the functions used. Two of them are unimodal (f_sph and f_sch) and four are multimodal (f_ack, f_ras, f_gri, f_ros). Two methods were used for DE: RAND/1/bin and BEST/1/bin.

Code	Name	Search range	f_bias
f_sph	Shifted Sphere	[-100,100]	-450
f_shc	Shifted Shwefel	[-100,100]	-450
f_ros	Shifted Rosenbrock	[-100,100]	390
f_ras	Shifted Rastrigrin	[-5,5]	-330
f_gri	Shifted Griewank	[-600, 600]	-180
f_ack	Shifted Ackley	[-32,32]	-140

Table 1. Functions used for the study.

The average error is defined as the difference between the current value of the global optimum and the value obtained by the algorithm. Table 2 shows the average error and the average standard deviation found with each method, discriminating the tests according to the dimension analyzed. Cells in bold indicate the best result. It can be observed that in most cases, when using any scheme of the hybrid *DE+LS* algorithm, better error and standard deviation values are obtained. Analyzing the performance of the proposal, it can be seen that better results are obtained with the *DE+LS(1)* method. In those instances in which the results are better using the classic DE version, the differences with the hybrid version are not significant. It can therefore be seen that the proposal demonstrates good performance based on the results of average error and standard deviation.

It also can be observed a correspondence by comparing the results obtained between the strategies RAND/1/bin and BEST/1/bin. If there was an improvement in the quality of the solutions using the BEST method (as opposed to RAND) with the classical DE version, that improvement is also registered

[3] Tests were performed with different values for F and Cr, using functions with similar features to those used in this work.

Func.- Met.	Dim 30		Dim 100		Dim 500	
f_sph Best	**Err.**	**Stdev.**	**Err.**	**Stdev.**	**Err.**	**Stdev.**
DE	4.029e-17	1.896e-17	**1.916e-16**	1.183e-16	2.522e-04	3.157e-04
DE+LS(1)	**3.760e-17**	1.962e-17	2.274e-16	1.409e-16	**1.212e-13**	**1.326e-13**
DE+LS(2)	3.849e-17	**1.755e-17**	2.175e-16	**1.058e-16**	2.152e-09	2.031e-09
f_sph Rand						
DE	0.000e+00	0.000e+00	2.286e-14	6.146e-15	1.241e+06	4.637e+04
DE+LS(1)	0.000e+00	0.000e+00	**2.453e-16**	**9.561e-17**	**7.358e+03**	**7.246e+02**
DE+LS(2)	0.000e+00	0.000e+00	3.993e-16	1.077e-16	1.409e+04	1.697e+03
f_sch Best						
DE	4.669e+02	7.713e+00	5.405e+02	6.213e+00	5.883e+02	3.380e+00
DE+LS(1)	**4.500e+02**	**1.565e-09**	**4.500e+02**	**1.638e-06**	**4.817e+02**	**1.990e+00**
DE+LS(2)	4.500e+02	1.462e-08	4.561e+02	1.924e+00	5.268e+02	2.016e+00
f_sch Rand						
DE	4.500e+02	5.940e-13	4.858e+02	2.100e+00	6.279e+02	1.895e+00
DE+LS(1)	**4.500e+02**	**6.002e-14**	**4.584e+02**	**0.846e+00**	5.843e+02	3.365e+00
DE+LS(2)	4.500e+02	1.048e-13	4.635e+02	1.344e+00	**5.499e+02**	**0.000e+00**
f_ros Best						
DE	**1.964e+01**	**1.077e+01**	2.174e+02	8.836e+01	7.022e+03	1.389e+04
DE+LS(1)	3.257e+01	2.396e+01	2.225e+02	1.187e+02	**7.690e+02**	**2.681e+02**
DE+LS(2)	2.261e+01	1.743e+01	**1.518e+02**	**8.305e+01**	2.232e+04	1.111e+05
f_ros Rand						
DE	3.391e+01	2.197e+01	1.625e+02	1.193e+02	1.838e+12	8.077e+10
DE+LS(1)	**2.292e+01**	**1.036e+01**	1.073e+02	**2.636e+01**	**1.054e+09**	**1.898e+08**
DE+LS(2)	3.204e+01	2.285e+01	**1.064e+02**	2.722e+01	2.976e+09	6.744e+08
f_ras Best						
DE	3.670e-17	1.602e-17	**2.032e-16**	**7.718e-17**	1.390e-06	2.771e-06
DE+LS(1)	**3.670e-17**	**1.602e-17**	2.130e-16	8.849e-17	**2.057e-14**	**2.380e-14**
DE+LS(2)	4.029e-17	1.896e-17	2.238e-16	1.095e-16	3.339e-11	6.686e-11
f_ras Rand						
DE	0.000e+00	0.000e+00	1.011e-16	7.618e-17	3.067e+03	8.827e+01
DE+LS(1)	0.000e+00	0.000e+00	5.461e-17	2.775e-17	**1.811e+01**	**1.332e+00**
DE+LS(2)	0.000e+00	0.000e+00	**5.372e-17**	**2.728e-17**	3.348e+01	3.129e+00
f_gri Best						
DE	3.497e-03	5.715e-03	5.226e-03	2.331e-02	1.990e-01	2.833e-01
DE+LS(1)	**1.668e-03**	**4.139e-03**	**2.384e-03**	**4.737e-03**	**4.039e-02**	**7.058e-02**
DE+LS(2)	4.768e-03	5.930e-03	3.733e-03	6.872e-03	3.546e-01	4.268e-01
f_gri Rand						
DE	0.000e+00	0.000e+00	1.196e-14	2.671e-15	1.062e+04	3.202e+02
DE+LS(1)	0.000e+00	0.000e+00	**1.383e-16**	**4.292e-17**	**6.057e+01**	**6.028e+00**
DE+LS(2)	0.000e+00	0.000e+00	2.395e-16	7.049e-17	1.145e+02	1.106e+01
f_ack Best						
DE	-2.718e+00	3.522e-15	-2.718e+00	**1.238e-14**	-2.716e+00	8.200e-03
DE+LS(1)	-2.718e+00	2.970e-15	**-2.718e+00**	9.867e-14	-2.718e+00	**2.997e-09**
DE+LS(2)	**-2.718e+00**	**2.858e-15**	-2.718e+00	1.643e-14	**-2.718e+00**	1.235e-07
f_ack Rand						
DE	-2.718e+00	0.000e+00	**-2.718e+00**	2.774e-09	1.631e+01	**6.616e-02**
DE+LS(1)	**-2.718e+00**	**0.00e+00**	-2.718e+00	**1.447e-10**	**1.240e+00**	1.759e-01
DE+LS(2)	-2.718e+00	0.000e+00	-2.718e+00	2.114e-10	2.373e+00	2.315e-01

Table 2. Average error and average standard deviation obtained with the proposed algorithm with each version.

for the BEST method in contrast to RAND in any of the two methods. This correspondence would indicate that the proposal is congruent among different methods of ED. If any of them reflects an improvement, the same occurs in each version of the proposed hybrid algorithm.

5.1 Statistical analysis: non parametric methods

In order to verify if there exists significant differences between the results obtained with the methods used, an analysis is given below using the Friedman paired test. The null hypothesis states that there are no significant differences between DE and the two hybrid versions proposed.

The average error results were the ones used to perform the test, over all problem sizes (30, 100, 500). Table 3 shows the obtained values. It can be observed for dimensions 30 and 100 (BEST method) that there is insufficient evidence to reject the null hypothesis, therefore it can not be ensured that there are significant differences between the algorithms compared for the stated problems. Looking again at table 2 it can be seen that the algorithms behavior for dimension 30 and for dimension 100 (BEST method) is more regular and balanced.

Problem size	**RAND**	**BEST**
30	0.529200	0.154200
100	0.009404	0.846500
500	0.005704	0.009404

Table 3. Friedman test. p-value obtained comparing *DE* vs *DE+LS(1)* vs *DE+LS(2)*.

However, in all comparisons made for dimension 500 a p-value < 0.05 is obtained, indicating that there is significant evidence to reject the null hypothesis. From this it is inferred that there are marked differences between the algorithms with all versions and methods applied. Looking at dimension 500 in table 2 for each function it can be analyzed that in general the performance is better using hybrid versions in contrast with the classic DE version. This is consistent with the results of the non-parametric test for this dimension, and for the dimension 100 using the Differential Evolution RAND method.

The trend would indicate that the larger the scale of the problems, differences become more evident. This assumption can be corroborated through the study and the test generation for larger problem dimensions

As stated above, and considering that in some cases there is a similarity in the quality of the solutions, the next section presents a study carried out to test whether the addition of LS improves the performance, in terms of the speed of convergence.

5.2 Convergence analysis

A convergence analysis was performed by calculating the average of the objective function evaluations needed to reach an error of 10^{-06}.

In the proposed algorithm there are three points where the objective function evaluation takes place. First, it is performed during the population initialization to calculate the initial costs. Next, in LS, the objective function is used to obtain the best neighbor, which is one population individual with the lowest cost in the function evaluation. Finally, in the selection step, the function used to evaluate the costs of the individuals resulting from mutation and crossover stages. To analyze the convergence of the proposed algorithm only the evaluations carried out in DE were taken into account, i.e., those corresponding to initialization and selection.

Table 4 shows the results obtained for those functions (using the respective methods). If the error 10^{-06} is not reached, the total number of evaluations is 600100.

Function	Method	Dim	DE	DE+LS(1)	DE+LS(2)
f_sph	BEST	30	27046	14403	16043
f_ras	BEST	30	20686	12200	12233
f_sph	BEST	100	82623	33576	51996
f_sph	RAND	100	361076	301360	309473
f_ras	BEST	100	64326	33576	40036
f_ras	RAND	100	281146	233020	237996
f_gri	RAND	100	353643	295580	302643
f_sph	BEST	500	600100	229126	431716
f_ras	BEST	500	600100	181230	349280

Table 4. Function evaluation average to obtain an error of 10^{-06}.

It can be seen that in all cases *DE+LS(1)* performs less objective function evaluations, in order to achieve the mentioned error.

6 Conclusions and future works

In this paper we made a comparison between the classic Differential Evolution algorithm and two hybrid version that combines DE with Local Search, in order to explore the solution space near the best population individual.

Through the results analysis it was found that the proposed algorithm gives no significan improvement using *DE+LS(2)*, but with the version *DE+LS(1)* the proposal obtains good quality results in contrast to the classical version of DE in terms of average error, average standard deviation and speed of convergence.

As future works we plan to carry out the study on more complex problems, an also to compare the proposal performance with other hybrid algorithm versions and with algorithms of the state of the art on the benchmark analyzed (for example, the algorithms from the CEC 2008 competition [11]).

References

1. M. Ali, M. Pant, A. Nagar. "Two local Search Strategies for Differential Evolution". IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications. ISBN:978-1-4244-6437-1. DOI:10.1109/BICTA.2010.5645285 2010
2. J. Brest, A. Zamuda, B. Bokovie, M. Maucec, Viljem Zumer. "High-Dimensional Real-Parameter Optimization using Self-Adaptive Differential Evolution Algorithm with Population Size Reduction". IEEE Congress on Evolutionary Computation (CEC 2008). ISBN:978-1-4244-1823-7. DOI:10.1109/CEC.2008.4631067. 2008.
3. F. Glover, G. Kochenberg. "Handbook of Metaheuristics". Kluwer Academic Publishers. 2003. ISBN: 1-4020-7263-5.
4. D. Johnson, L. McGeoch. "The Traveling Salesman Problem: A Case Study in Local Optimization". In "Local Seach in Combinatorial Optimization". Editorial Aarts and J. k. Lenstra. 1995.
5. H. Lourenco. "Iterated Local Search". Chapter 11. "Handbook of Metaheuristics" by Michel Gendreau and Jean Yves Potvin. 2010. International Series in Operations Research and Management Science, Vol. 146.
6. C. Martınez, F. Rodriguez, M. Lozano. "Role differentiation and malleable mating for differential evolution: an analysis on large-scale optimisation". Soft Computing 15:11 (2011) p:2109-2126. DOI:10.1007/s00500-010-0641-8
7. N. Noman, H. Iba. "Accelerating Differential Evolution Using an Adaptive Local Search". IEEE Transactions on Evolutionary Computation. ISSN: 1089-778X. DOI:10.1109/TEVC.2007.895272. 2008.
8. I. Osman, J. Kelly "Meta-Heuristics: Theory & Applications". Springer. 1996. ISBN: 978-0792397007.
9. K. Price, R. Stron, J. Lampinen. "Differential Evolution: A Practical Approach to Global Optimization". Springer. ISBN: 3-540-20950-6. 2005.
10. E. Talbi. "Metaheuristics: From Design to Implementation". 2009 John Wiley & Sons, Inc.
11. K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. P. Chen, C. M. Chen, and Z. Yang, "Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization", Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, 2007.
12. Z. Yang, K. Tang, X. Yao. "Self-adaptative Differential Evolution with Neighborhood Search". IEEE Congress on Evolutionary Computation (CEC 2008). p:1110-1116. ISBN:978-1-4244-1823-7 .
13. Z. Yang, K. Tang, X. Yao. "Scalability of generalized adaptative differential evolution for large-scale continuous optimization." Soft Computing, 15:11, p: 2141-2155 (2011). DOI 10.1007/s00500-010-0643-6.
14. A. Zamuda, J. Brest, B. Bokovie , V. Zumer. "Large Scale Global Optimization Using Differential Evolution With Self Adaptation and Cooperative Co-evolution". IEEE Congress on Evolutionary Computation (CEC 2008). p:1110-1116. ISBN:978-1-4244-1823-7. DOI:10.1109/CEC.2008.4631301.
15. S. Zhao, J. Liang, P. Suganthan, M. Tasgetiren. "Dynamic Multi-Swarm Particle Swarm Optimizer with Local Search for Large Scale Global Optimization". ISBN:978-1-4244-1823-7. 2008
16. S. Zhao, P. Suganthan, S. Das. "Self-adaptative differential evolution with multi-trajectory search for large-scale optimization". Soft Computing - A Fusion of Foundations, Methodologies and Applications (Septiembre 2010), Vol. 15. Nro 11. pp. 1-11-11, DOI:10.1007/s00500-010-0645-4.