
# PERFORMANCE ANALYSIS OF A PARTICLE SWARM OPTIMIZATION ALGORITHM TO SOLVE MULTIOBJECTIVE OPTIMIZATION PROBLEMS

**Érica C.R. Carvalho**

**Afonso C.C. Lemonge**

**José P.G. Carvalho**

**Patrícia H. Hallak**

**Heder S. Bernardino**

ericacrcarvalho@gmail.com

afonso.lemonge@ufjf.edu.br

jose.carvalho@engenharia.ufjf.br

patricia.hallak@ufjf.edu.br

heder@ice.ufjf.br

Federal University of Juiz de Fora, José Lourenço st, Juiz de Fora, MG - Brazil 36036-330

***Abstract.*** *The interest in multiobjective optimization algorithms has grown in recent years due to its applicability in problems from several areas, especially those from engineering. In general, the objectives considered in these problems are conflicting and a Pareto Front curve composed by the non-dominated solutions is searched as the expected solution. In the context of evolutionary computation there are many of algorithms applied to this type of problem, such as genetic algorithms, differential evolution, and particle swarm. In addition, multiobjective problems may present constraints turning them more complex, and requiring effective strategies that can direct the search for solutions that are in the feasible search space. This paper aims to evaluate the ability of a multiobjective optimization algorithm, called Multiobjective Craziness based Particle Swarm Optimization (MOCRPSO) in a set of unconstrained and constrained benchmark functions. An Adaptive Penalty Method (APM), which has been successfully applied to solving mono and multiobjective optimization problems, is used to handle the constraints. The results found by MOCRPSO illustrate the efficiency of the algorithm when compared to the results found in the literature.*

***Keywords:*** *Particle Swarm Optimization, Multiobjective Optimization, Adaptive Penalty Method*

# 1 INTRODUCTION

The simultaneous optimization of more than one objective function is defined as multiobjective optimization (Deb, 2001). Evolutionary Algorithms (EAs), specially the population-based metaheuristics, have becoming attractive approaches used to solve multiobjective optimization problems. When these algorithms are adopted, a set of Pareto optimal solutions can be obtained in a single run.

Several EAs have been developed for solving multiobjective optimization problems. Strength Pareto Evolutionary Algorithm 2 (SPEA2) have been proposed by (Laumanns, 2001). Schaffer (1985) proposed a Genetic Algorithm Multiobjective called Vector Evaluated Genetic Algorithm (VEGA). Deb *et al.* (2002) proposed a Non-dominated Sorting Genetic Algorithm II (NSGA-II) that is a version of NSGA Non-dominated Sorting Genetic Algorithm (NSGA) (Srinivas and Deb, 1994). (Ellaia *et al.*, 2013) proposed the Accelerated MO Particle Swarm Optimization (AMOPSO).

Particle Swarm Optimization (PSO) is a metaheuristic introduced by Eberhart and Kennedy (1995) and widely used in the literature as the Genetic Algorithm (GA) (Holland, 1973), Ant Colony Optimization (ACO) (Colorni *et al.*, 1991), Artificial Bee Colony (ABC) (Majumdar *et al.*, 2012), among others. In th this paper a modified version of PSO called Multiobjective Craziness-based Particle Swarm Optimization (CRPSO) and proposed by (Kar *et al.*, 2012) is used. An Adaptive Penalty Method (APM) is coupled to the proposed algorithm to handle the constraints. No attempt to survey the current literature is conducted in this paper and the reader can be referred to related papers in this field.

The numerical experiments consist in six unconstrained and constrained test-functions traditionally found in the literature (Deb *et al.*, 2002) which are used to comparatively evaluate the performance of the MOCRPSO. The results are compared with those found by the Non-dominated Sorting Genetic Algorithm (NSGA-II). Two different performance metrics, namely the Empirical Attainment Function (EAF) (Fonseca and Fleming, 1996) and Hypervolume (Zitzler and Thiele, 1999), are used here.

The paper is organized as follows. Section 2 presents a formulation of a multiobjective optimization problem. Section 3 describes the multiobjective algorithm adopted and the constraint-handling technique used. The computational experiments are presented in Section 4 and, finally, the paper ends with the conclusions in Section 5.

# 2 MULTIOBJECTIVE OPTIMIZATION

In Single-objective Optimization (SO) the task is to find one solution which optimizes the sole objective function (Deb, 2001). On the other hand, in Multiobjective Optimization (MO) there is no single optimal solution, and the optimal decisions need to be taken in the presence of trade-off between two or more conflicting objectives.

Multiobjective Optimization Problems (MOPs) are very common in many areas including engineering (Angelo *et al.*, 2015), economics (Sabio *et al.*, 2014), logistics (Pereira *et al.*, 2017), among others. A MOP has a number of objective functions which are to be minimized (or maximized). A general Multiobjective Optimziation Problem can be formulated as (Angelo *et al.*, 2015)

minimize

$$f_i(x), f_2(x), ..., f_k(x) \tag{1}$$

subject to

$$x_j^L \leq x_j \leq x_j^U,\ j = 1, ..., n,$$

where $k$ ($\geq 2$) is the number of objective functions to be minimized simultaneously, $x$ is the decision variable, $x^L$ and $x^U$ are the lower and upper bounds and $n$ is the number of decisions variables.

The concept of dominance is used in most multiobjective optimization algorithms (Deb, 2001). The set of solutions of a MOP consists of all decision vectors for which the corresponding objective vectors cannot be improved in any dimension without degradation in another. These vectors are known as Pareto optimal (Zitzler and Thiele, 1999). A solution $x_1 \in X$ dominates a solution $x_2 \in X$ if, and only if $x_1$ is not worse than $x_2$ for all the objectives and $x_1$ is better than $x_2$ for at least one of the objectives. Thus, a solution is said to be non-dominated if it is impossible to improve one component of the solution without worsening the value of at least one other component of the solution (Raquel and Naval Jr, 2005).

The goals of Multiobjective Optimization are (Raquel and Naval Jr, 2005):

- to guide the search toward the true Pareto frot (image of non-dominated solutions in the objective space) or approximate the Pareto optimal set and
- to generate a well-distributed Pareto front.

# 3 THE MOCRPSO ALGORITHM AND THE PENALTY SCHEME

## 3.1 The baseline MOCRPSO algorithm

Particle Swarm Optimization (PSO) is a metaheuristic introduced by Eberhart and Kennedy (1995) and widely used in the literature. The algorithm based on the flocking and schooling patterns of birds flocking and the individuals, referred to as particles, "flown" through the search space. A modified version of PSO called Craziness-based Particle Swarm Optimization (CRPSO) and proposed by Kar *et al.* (2012) is used in this paper. The velocity expression $v_i$ and the position $x_i$ can be expressed as follows (Kar *et al.*, 2012):

$$\begin{aligned} v_j^{(i)}(t+1) = \quad & r_2 \cdot sign(r_3) \cdot v_j^{(i)}(t) + (1-r_2)c_1 \cdot r_1(x_{pbest}^{(i)} - x_j^{(i)}) + (1-r_2) \cdot c_2 \cdot \\ & (1-r_1)(x_{gbest} - x_j^{(i)}) + P(r_4) \cdot sign2(r_4) \cdot v_j^{craziness} \end{aligned} \tag{2}$$

$$x_j^{(i)}(t+1) = x_j^{(i)}(t) + v_j^{(i)}(t+1) \tag{3}$$

where $r_1$, $r_2$, $r_3$ and $r_4$ are the random parameters uniformly taken from the interval [0,1), $sign(r_3)$ is a function defined as

$$sign(r_3) = \begin{cases} -1, & r_3 \leq 0.5 \\ 1, & r_3 > 0.5 \end{cases} \tag{4}$$

$v_j^{craziness}$, the craziness velocity, is a user defined parameter from the interval [$v^{min}$, $v^{max}$] and $P(r_4)$, $sign2(r_4)$ are defined, respectively, as

$$P(r_4) = \begin{cases} 1, & r_4 \leq Pcr \\ 0, & r_4 > Pcr \end{cases} \tag{5}$$

$$sign2(r_4) = \begin{cases} -1, & r_4 \geq 0.5 \\ 1, & r_4 < 0.5 \end{cases} \tag{6}$$

and $Pcr$ is a predefined probability of craziness. Although the parameter $Pcr$ is fixed, $P(r_4)$ is defined every time the velocity is calculated.

The multiobjective version of CRPSO algorithm called Multiobjective Craziness based Particle Swarm Optmization (MOCRPSO) is based on the MOPSO-CD[1] algorithm proposed by Raquel and Naval Jr (2005). The algorithm uses the concept of an external archive $A$ to store the non-dominated solutions. The new velocity of the MOCRPSO can be expressed as

$$\begin{aligned} v_j^{(i)}(t+1) = \quad & r_2 \cdot sign(r_3) \cdot v_j^{(i)}(t) + (1-r_2)c_1 \cdot r_1(x_{pbest}^{(i)} - x_j^{(i)}) + (1-r_2) \cdot c_2 \cdot \\ & (1-r_1)(A[gbest] - x_j^{(i)}) + P(r_4) \cdot sign2(r_4) \cdot v_j^{craziness} \end{aligned} \tag{7}$$

Some mechanism are incorporated in MOCRPSO to maintain the diversity of non-dominated solutions (Raquel and Naval Jr, 2005): crowding distance, global best selection and mutation. The crowding distance is incorporated into the algorithm specifically on global best selection and in the external archive. Fig. 1 shows the calculation of the crowding distance of point $i$, which is an estimate of the size of the largest cuboid enclosing $i$ without including any other point (Raquel and Naval Jr, 2005).

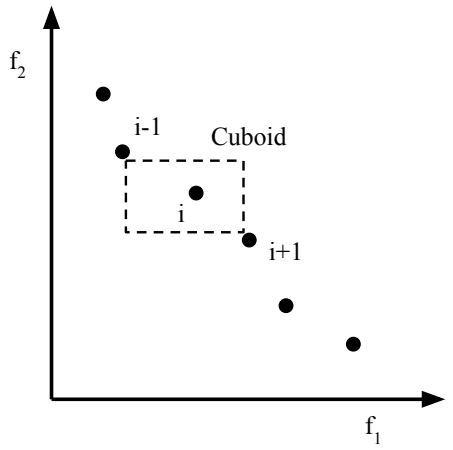


**Figure 1: Crowding distance, adapted from (Raquel and Naval Jr, 2005).**

[1]https://sites.google.com/site/prosnaval/codes

The global best of the particles is selected here from those non-dominated solutions with the highest crowding distance values. Whenever the archive is full, crowding distance is again used in the selection of the solution that should be replaced in the external archive (Raquel and Naval Jr, 2005). An operator of mutation is also used in the algorithm to increase diversity in the swarm. This is helpful in terms of preventing premature convergence due to existing local Pareto fronts in some optimization problems (Raquel and Naval Jr, 2005).

The steps of the MOCRPSO algorithm are presented in Algorithm 1.

**Algorithm 1:** Pseudocode of the MOCRPSO algorithm.

**for** $i \leftarrow 0$ *to* $sizeSwarm$ **do**
    initialize $x_i$ and $v_i$ randomly ($x$ is the swarm of particles);
    evaluate $x_i$;
    initialize $pBest_i$;
**end**
initialize $gBest$;
store the non-dominated vectors found in $x$ into $A$ ($A$ is the external archive that stores non-dominated solutions);
**repeat**
    compute the crowding distance values of each non-dominated solution in the archive $A$;
    sort the non-dominated solutions in $A$ in descending crowding distance values;
    **for** $i \leftarrow 0$ *to* $sizeSwarm$ **do**
        randomly select the global best guide for $x_i$ from a specified top portion of the sorted archive $A$ and store its position to $gBest$.;
        update $x_i$ according to Equation 3;
        update $v_i$ according to Equation 7;
        **if** *t < mutationRate* **then**
            perform mutation on $x_i$;
        **end**
        evaluate $x_i$;
        $pBest_i \leftarrow$ best between $x_i$ e $pBest_i$ ;
    **end**
    insert all new non-dominated solution in $x$ into $A$ if they are not dominated by any of the stored solutions (All dominated solutions in the archive by the new solution are removed from the archive);
    **if** $A$ *is full* **then**
        compute the crowding distance values of each non-dominated solution in the archive $A$;
        sort the non-dominated solutions in $A$ in descending crowding distance values;
        randomly select a particle from a specified bottom portion which comprise the most crowded particles in the archive then replace it with the new solution;
    **end**
**until** *maximum number of iterations is reached*;

## 3.2 Handling constrained problems

Lemonge and Barbosa (2004) proposed a method to handle constraints called Adaptive Penalty Method (APM). This method is adopted here to enforce all the constraints of the test-functions in the numerical experiments. Considering each objective separately, the fitness function $F(x)$ can be written as

$$F(x) = \begin{cases} f(x), & \text{if} \quad x \quad \text{is feasible} \\ \bar{f}(x) + \sum_{j=1}^{n_c} k_j v_j(x), & \text{otherwise} \end{cases} \tag{8}$$

and

$$\bar{f}(x) = \begin{cases} f(x), & \text{if} \quad f(x) > \langle f(x) \rangle \\ \langle f(x) \rangle, & \text{if} \quad f(x) \leq \langle f(x) \rangle \end{cases} \tag{9}$$

where $\langle f(x) \rangle$ is the average value of the objective function of the current population. The penalty parameter $k_j$ is defined as

$$k_j = | \langle f(x) \rangle | \frac{\langle v_j(x) \rangle}{\sum_{l=1}^{n_c} [\langle v_l(x) \rangle]^2} \tag{10}$$

where $\langle v_j(x) \rangle$ means the violation of the j-$th$ constraint averaged over the current population considering only unfeasible individuals.

# 4 COMPUTATIONAL EXPERIMENTS

In this section, the results obtained by MOCRPSO are compared with those obtained by NSGA-II[2] (Deb *et al.*, 2002). The parameters used in MOCRPSO are: $c_1 = c_2 = 2.05$, $v_j^{craziness} = 0.001$ and $Pcr = 0.5$. In all experiments, the maximum number of objective function evaluations is 50000 and the number of independent runs is 100. Each experiment uses a different randomly generated initial population. The Multiobjective Optimization Problems solved here consists in six test-functions. The objective functions and the constraints of each function are described below.

## 4.1 Test-functions

The first set of test-functions is Zitzler-Deb-Thiele - ZDT family of functions (Zitzler *et al.*, 2000). It is composed of six problems (ZDT1 - ZDT6) based on the above construction process. These function are minimization problems and have two objective functions. In this paper, three of ZDT's function was used and can be described as follows

[2]http://www.egr.msu.edu/ kdeb/codes.shtml

**ZDT1**

minimize

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= a(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \sqrt{\frac{f_1(x)}{a(x)}} \end{aligned} \tag{11}$$

**ZDT2**

minimize

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= a(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \left(\frac{f_1(x)}{a(x)}\right)^2 \end{aligned} \tag{12}$$

**ZDT3**

minimize

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= g(x)b(f_1(x), a(x)) \\ a(x) &= 1 + \frac{9}{29}\sum_{i=2}^{30} x_1 \\ b(f_1(x), a(x)) &= 1 - \sqrt{\frac{f_1(x)}{a(x)}} - \left(\frac{f_1(x)}{a(x)}\right)^2 \sin(10\pi f_1(x)) \end{aligned} \tag{13}$$

The second test-function is known as CTP1. This function has two objective functions and two constraints and can be describe as follows (Deb, 2001)

minimize

$$\begin{aligned} f_1(x) &= x \\ f_2(x) &= (1 + x_2)\exp\left(\frac{x_1}{1 + x_2}\right) \end{aligned}$$

subjec to

$$\begin{aligned} g_1(x) &= \frac{f_2(x)}{0.858\exp(-0.541 f_1(x))} \geq 1 \\ g_2(x) &= \frac{f_2(x)}{0.728\exp(-0.295 f_1(x))} \geq 1 \end{aligned} \tag{14}$$

The third test-function used contains two objective functions and six constraints. The function is defined as

minimize

$$f_1(x) = -25(x_1-2)^2 - (x_2-2)^2 - (x_3-1)^2 - (x_4-4)^2 - (x_5-1)^2)$$

$$f_2(x) = \sum_{i=1}^{6} x_i^2$$

subjecto to

$$\begin{aligned} g_1(x) &= x_1 + x_2 - 2 \geq 0 \\ g_2(x) &= 6 - x_1 - x_2 \geq 0 \\ g_3(x) &= 2 - x_2 + x_1 \geq 0 \\ g_4(x) &= 2 - x_1 + 3x_2 \geq 0 \\ g_5(x) &= 4 - (x_3-3)^2 - x_4 \geq 0 \\ g_6(x) &= (x_5-3)^2 + x_6 - 4 \geq 0 \end{aligned} \tag{15}$$

The fourth and last test-function is composed of three objective functions and there are no constraints. Viennet function is described as follows

minimize

$$\begin{aligned} f_1(x) &= 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \\ f_2(x) &= \frac{(3x_1 - 2x_2 + 4)^2}{8} + \frac{(x_1 - x_2 + 1)^2}{27} + 15 \\ f_3(x) &= \frac{1}{x_1^2 + x_2^2 + 1} 1.1\exp(-(x_1^2 + x_2^2)) \end{aligned} \tag{16}$$

## 4.2 Performance Metrics

In the multiobjective optimization, it is often impossible to know the Pareto optimal set. In these conditions, to measure the performance of the algorithm it is not an easy task (Nicoară, 2007). Various performance metrics for Multiobjective Evolutionary Algorithms (MOEAs) can be found in references (Deb, 2001) (Nicoară, 2007) (Vargas *et al.*, 2015). In this paper two performance metrics are used: Empirical Attainment Function and Hypervolume.

The notion of attainment function was first introduced by Fonseca and Fleming (1996) and studies the distribution of a random non-dominated point by providing information about the probability of a given point being weakly dominated. This probability can be estimated from

several runs of an algorithm. The Empirical Attainment Function - EAF can be estimated as (Grunert da Fonseca *et al.*, 2001)

$$\alpha_n(z) = \frac{1}{n}\sum_{i=1}^{n} b_i(z) \tag{17}$$

where $z \in \mathbb{R}^d$ and $b_1(z), ..., b_n(z)$ are $n$ realizations of the attainment indicator $b_x(z)$.

López-Ibánez *et al.* (2010) developed a tool for graphical analysis of the EAF[3]. Fig. 2 shows three EAF curves (best, median and worst cases) obtained by 10 independent runs of a given MOEA on a problem for two objective functions. The best curve delimits the region dominated by all non-dominated solutions obtained in the 10 runs. The median EAF curve represents 50% of the attainment surface. Finally, the worst curve bounds the region dominated by any non-dominated solution found by the search technique in the 10 runs. Details about EAF code can be found in (Fonseca and Fleming, 1996) and (Coello *et al.*, 2007).

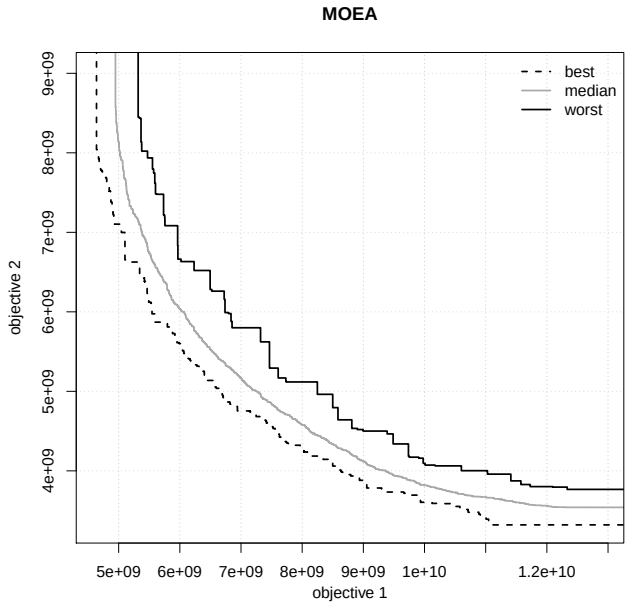


**Figure 2: Example of the EAF curves (best, median and worst cases).**

The second performance metric is defined as Hypervolume (Zitzler and Thiele, 1999) and provides a qualitative measure of convergence as well as diversity in a combined sense. It allows an easy comparison between the results of different multiobjective algorithms, by assigning a real value to a set of points. Mathematically, according to Deb (2001), for each solution $i \in Q$, a hypercube $v_i$ is obtained with a reference point $W$ and the solution $i$ as the diagonal corners of the hypercube. The reference point can simply be found by constructing a vector of worst objective function values. Thereafter, an union of all hypercubes is found and its hypervolume

[3]http://lopez-ibanez.eu/eaftools

(HV) is calculated

$$\text{HV} = \text{volume}\left(\cup_{i=1}^{|Q|} v_i\right) \tag{18}$$

## 4.3 Results

The EAF curves (best, median and worst) of the MOCRPSO and NSGA-II algorithms in the analyzed problems (except to Viennet function) are presented in Figs. 3 to 7. As the tool used here to generate the EAF curves does not support three-dimensional functions, the EAF curves are not be displayed for Viennet function. The comparisons of the algorithms are presented by differences between the best EAF, that represents the image of the non-dominated solutions obtained in all independent runs, and can be seen in Figs. 8 to 13. The simple three-dimensional EAF code can be found in (Fonseca *et al.*, 2011)[4]. Also, the normalized Hypervolumes are presented in these figures for all test-functions.

One can observe that the EAF curves of the solutions found by MOCRPSO are similar to those obtained by NSGA-II. The Hypervolume obtained by MOCRPSO from the first four test-functions (ZDT1, ZDT2, ZDT3 and CTP1) are larger than those obtained by NSGA-II. In Osyezka Kundu and Viennet functions, MOCRPSO obtained smaller but competitive Hypervolume values.

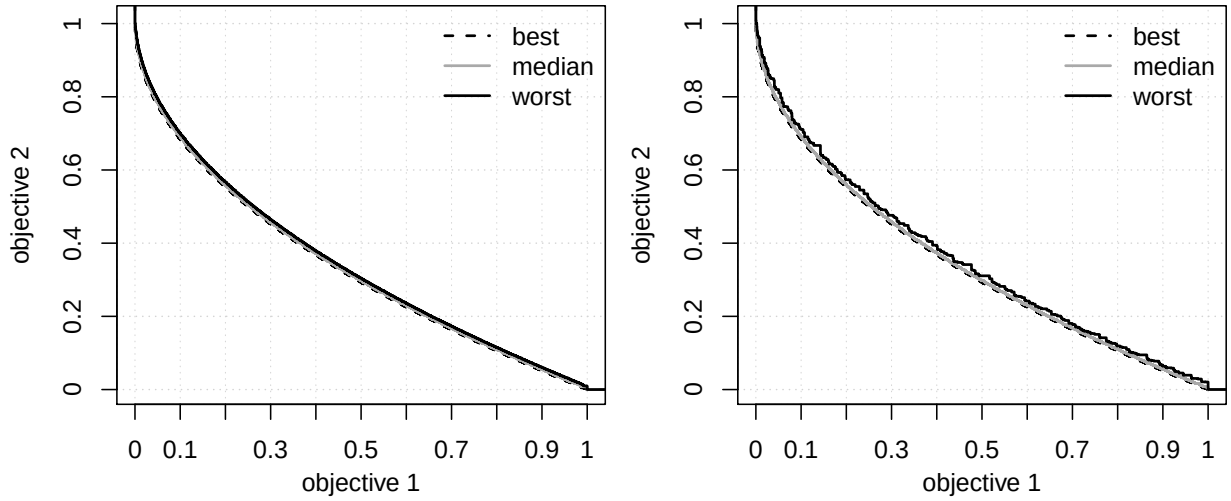


**Figure 3: EAF curves (best, median and worst) of the MOCRPSO (left) and NSGA-II (right) for the ZDT1 function.**

[4]https://eden.dei.uc.pt/ cmfonsec/software.html

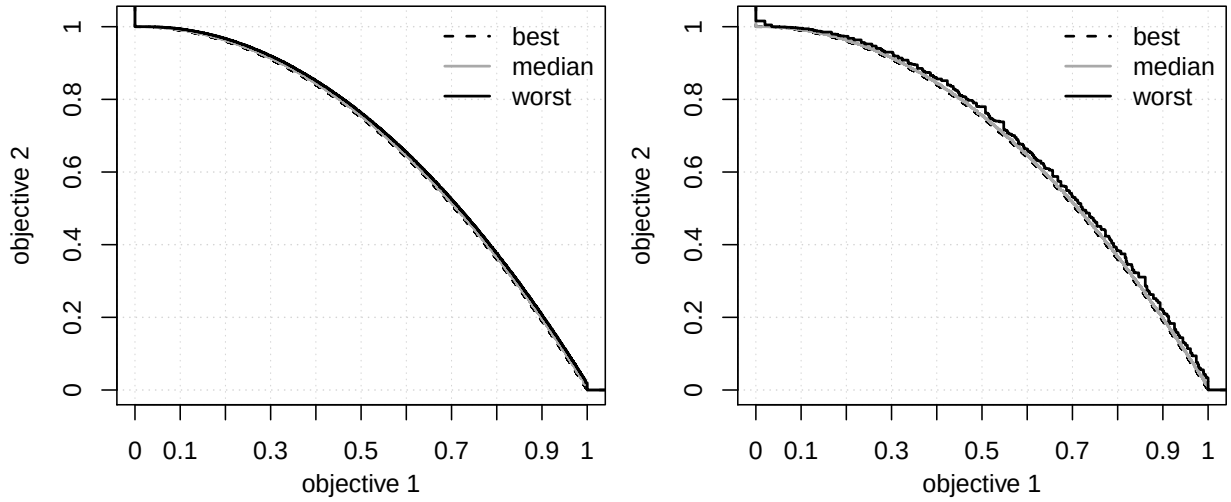


**Figure 4: EAF curves (best, median and worst) of the MOCRPSO (left) and NSGA-II (right) for the ZDT2 function.**

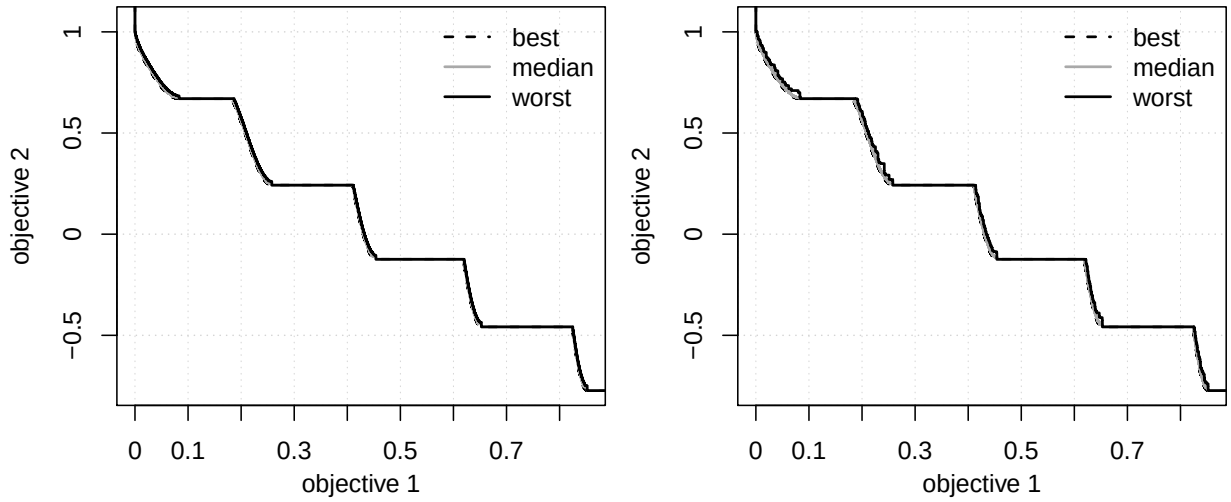


**Figure 5: EAF curves (best, median and worst) of the MOCRPSO (left) and NSGA-II (right) for the ZDT3 function.**

# 5 CONCLUSIONS

This paper evaluated the performance of a Multiobjective Craziness based Particle Optimization algorithm (MOCRPSO) in a set of unconstrained and constrained benchmark functions. An Adaptive Penalty Method (APM) is used to handle the constraints. The results are

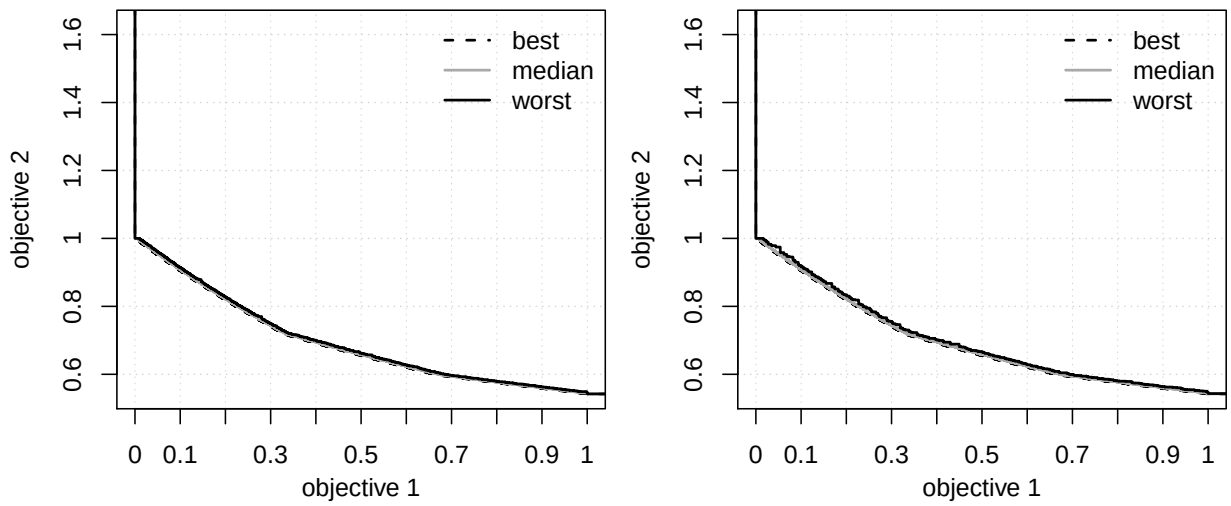


**Figure 6: EAF curves (best, median and worst) of the MOCRPSO (left) and NSGA-II (right) for the CTP1 function.**

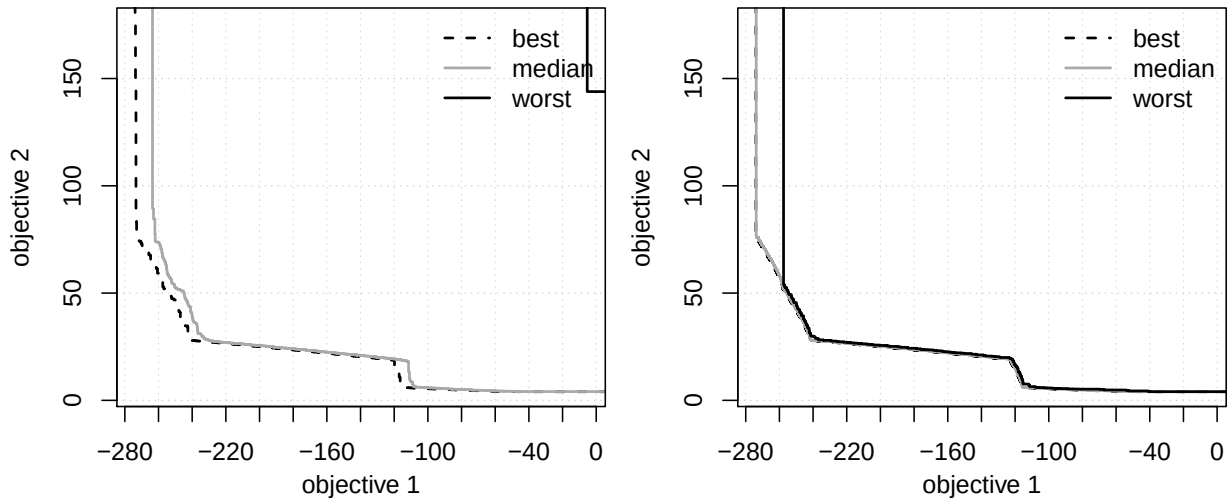


**Figure 7: EAF curves (best, median and worst) of the MOCRPSO (left) and NSGA-II (right) for the Osyczka Kundu function.**

compared to those found by the Non-dominated Sorting Genetic Algorithm (NSGA-II). The Empirical Attainment Function (EAF) and the Hypervolume are adopted here to analyze the results.

The results found by MOCRPSO shown to be competitive when compared to those found

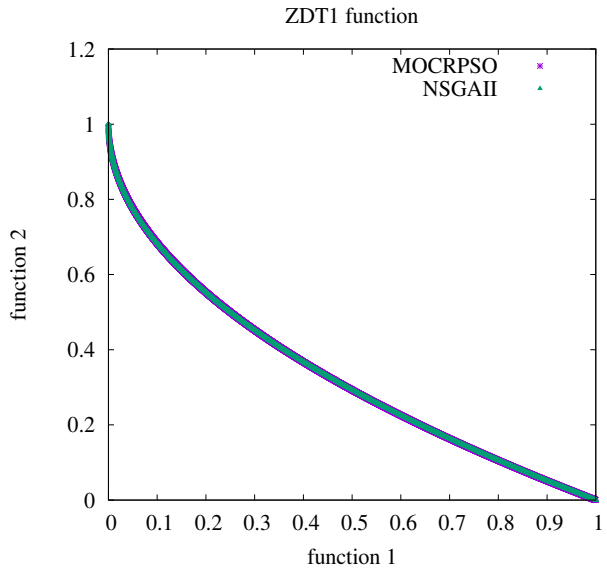


**Figure 8: Differences between the best EAF curve of the MOCRPSO and NSGA-II algorithms for the ZDT1 function. Normalized Hypervolume MOCRPSO: 0.666661 and NSGA-II: 0.666503.**

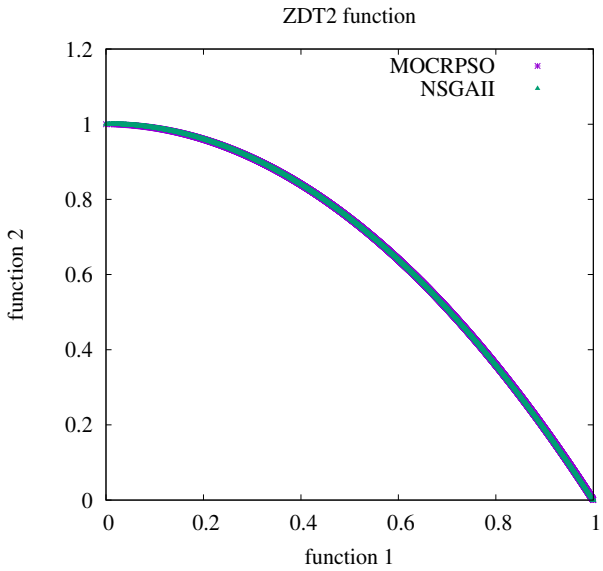


**Figure 9: Differences between the best EAF curve of the MOCRPSO and NSGA-II algorithms for the ZDT2 function. Normalized Hypervolume MOCRPSO: 0.333322 and NSGA-II: 0.333190.**

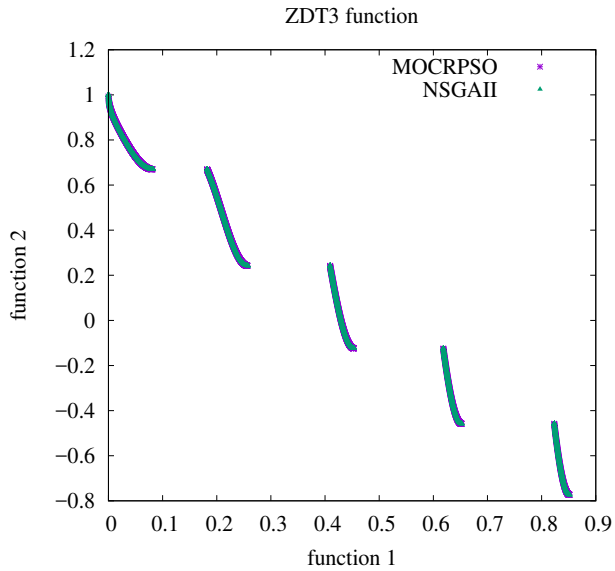


**Figure 10: Differences between the best EAF curve of the MOCRPSO and NSGA-II algorithms for the ZDT3 function. Normalized Hypervolume MOCRPSO: 0.517434 and NSGA-II: 0.517388.**

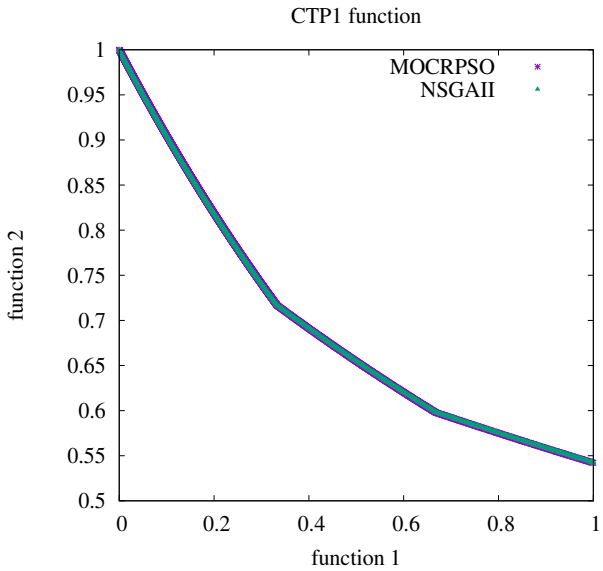


**Figure 11: Differences between the best EAF curve of the MOCRPSO and NSGA-II algorithms for the CTP1 function. Normalized Hypervolume MOCRPSO: 0.672642 and NSGA-II: 0.672413.**

by NSGA-II. In the six test-fucntions used in the experiments, MOCRPSO obtained greater or competitive Hypervolume values. For future works, the proposed algorithm will be applied for solving more complex multiobjective optimization problems from the real world, such as those from the structural and mechanical engineering.

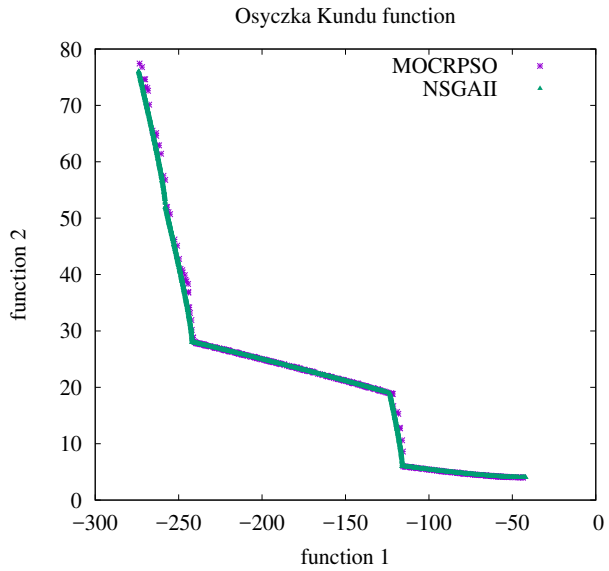


**Figure 12: Differences between the best EAF curve of the MOCRPSO and NSGA-II algorithms for the Osyczka Kundu function. Normalized Hypervolume MOCRPSO: 755558 and NSGA-II: 0.763134.**

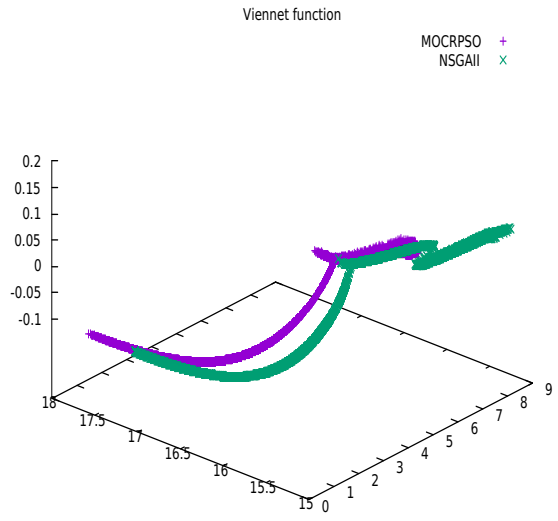


**Figure 13: Pareto front of the MOCRPSO and NSGA-II algorithms for the Viennet function. Normalized Hypervolume MOCRPSO: 0.756815 and NSGA-II: 0.874023.**

## ACKNOWLEDGEMENTS

The authors wish to thank the referees that helped the quality of the paper, CNPq (305175/2013-0 e 305099/2014-0), FAPEMIG (TEC PPM 528/11, TEC PPM 388/14 and TEC APQ 00103-12) and CAPES for their support.

## REFERENCES

Angelo, J. S., Bernardino, H. S., and Barbosa, H. J., 2015. Ant colony approaches for multiobjective structural optimization problems with a cardinality constraint. *Advances in Engineering Software*. vol. 80, pp. 101–115.

Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A., *et al.*, 2007. *Evolutionary algorithms for solving multi-objective problems*. vol. 5. Springer.

Colorni, A., Dorigo, M., and Maniezzo, V., 1991. Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*. vol. 142. Paris, France. pp. 134–142.

Deb, K., 2001. *Multi-objective optimization using evolutionary algorithms*. vol. 16. John Wiley & Sons.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*. vol. 6, n. 2, pp. 182–197.

Eberhart, R. and Kennedy, J., 1995. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. IEEE. pp. 39–43.

Ellaia, R., Habbal, A., and Pagnacco, E., 2013. A New Accelerated Multi-objective Particle Swarm Algorithm. Applications to Truss Topology Optimization. In *10th World Congress on Structural and Multidisciplinary Optimization*.

Fonseca, C. M. and Fleming, P. J., 1996. On the performance assessment and comparison of stochastic multiobjective optimizers. In *International Conference on Parallel Problem Solving from Nature*. Springer. pp. 584–593.

Fonseca, C. M., Guerreiro, A. P., López-Ibáñez, M., and Paquete, L., 2011. On the computation of the empirical attainment function. In *International Conference on Evolutionary Multi-Criterion Optimization*. Springer. pp. 106–120.

Grunert da Fonseca, V., Fonseca, C., and Hall, A., 2001. Inferential performance assessment of stochastic optimisers and the attainment function. In *Evolutionary multi-criterion optimization*. Springer. pp. 213–225.

Holland, J. H., 1973. Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing*. vol. 2, n. 2, pp. 88–105.

Kar, R., Mandal, D., Mondal, S., and Ghoshal, S. P., 2012. Craziness based Particle Swarm Optimization algorithm for FIR band stop filter design. *Swarm and Evolutionary Computation*.

Laumanns, M., 2001. *SPEA2: Improving the strength Pareto evolutionary algorithm*. Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK).

Lemonge, A. C. and Barbosa, H. J., 2004. An adaptive penalty scheme for genetic algorithms in structural optimization. *International Journal for Numerical Methods in Engineering*. vol. 59, n. 5, pp. 703–736.

López-Ibánez, M., Paquete, L., and Stützle, T., 2010. Exploratory analysis of stochastic local search algorithms in biobjective optimization. In *Experimental methods for the analysis of optimization algorithms*. Springer. pp. 209–222.

Majumdar, A., Maiti, D. K., and Maity, D., 2012. Damage assessment of truss structures from changes in natural frequencies using ant colony optimization. *Applied Mathematics and Computation*. vol. 218, n. 19, pp. 9759–9772.

Nicoară, E. S., 2007. Performance Measures for Multi-objective Optimization Algorithms. *Buletinul Universităţii Petrol–Gaze din Ploieşti., Seria Matematică-Informatică-Fizică*. vol. 59, n. 1, pp. 19–28.

Pereira, A. A. S., Barbosa, H. J., and Bernardino, H. S., 2017. Predator-Prey Techniques for Solving Multiobjective Scheduling Problems for Unrelated Parallel Machines. In *International Conference on Evolutionary Multi-Criterion Optimization*. Springer. pp. 484–498.

Raquel, C. R. and Naval Jr, P. C., 2005. An effective use of crowding distance in multiobjective particle swarm optimization. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. ACM. pp. 257–264.

Sabio, N., Pozo, C., Guillén-Gosálbez, G., Jiménez, L., Karuppiah, R., Vasudevan, V., Sawaya, N., and Farrell, J. T., 2014. Multiobjective optimization under uncertainty of the economic and

life-cycle environmental performance of industrial processes. *AIChE Journal*. vol. 60, n. 6, pp. 2098–2121.

Schaffer, J. D., 1985. Some experiments in machine learning using vector evaluated genetic algorithms. Tech. rep.. Vanderbilt Univ., Nashville, TN (USA).

Srinivas, N. and Deb, K., 1994. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary computation*. vol. 2, n. 3, pp. 221–248.

Vargas, D. E. d. C. *et al.*, 2015. Um algoritmo de evolução diferencial com penalização adaptativa para otimização estrutural multiobjetivo.

Zitzler, E., Deb, K., and Thiele, L., 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*. vol. 8, n. 2, pp. 173–195.

Zitzler, E. and Thiele, L., 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE transactions on Evolutionary Computation*. vol. 3, n. 4, pp. 257–271.