A Particle Swarm Optimizer for Multi-Objective Optimization

Leticia Cagnina, Susana Esquivel
Lab. de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)*
Universidad Nacional de San Luis - Ej. de los Andes 950 - (5700) San Luis, Argentina
{*lcagnina, esquivel*}*@unsl.edu.ar*

Carlos A. Coello Coello
CINVESTAV-IPN (Evolutionary Computation Group)
Electrical Eng. Department - Av. IPN No. 2508, Col. - México D.F. 07300, MÉXICO
ccoello@cs.cinvestav.mx

ABSTRACT

This paper proposes a hybrid particle swarm approach called Simple Multi-Objective Particle Swarm Optimizer (SMOPSO) which incorporates Pareto dominance, an elitist policy, and two techniques to maintain diversity: a mutation operator and a grid which is used as a geographical location over objective function space.
In order to validate our approach we use three well-known test functions proposed in the specialized literature.
Preliminary simulations results are presented and compared with those obtained with the Pareto Archived Evolution Strategy (PAES) and the Multi-Objective Genetic Algorithm 2 (MOGA2). These results also show that the SMOPSO algorithm is a promising alternative to tackle multi-objective optimization problems.

Keywords: Particle Swarm Optimization, Multi-objective Optimization, Pareto Optimality.

1. INTRODUCTION

Problems with multiple objectives are present in a great variety of real-life optimization problems. In these problems there are several conflicting objectives to be optimized and it is difficult to identify what the *best solution* is.

Despite the considerable diversity of techniques developed in the Operations Research field to tackle these problems, their intrinsic complexity calls for alternative approaches. Over the last decades, heuristics that find approximate solutions have attracted great interest. From these heuristics the Multi-Objective Evolutionary Algorithms (MOEAs) have been found to be very successful to solve multi-objective optimization problems [1], [2], [3], [4], [5].

Another technique that has been adopted in the last years for dealing with multi-objective optimization problems is Particle Swarm Optimization (PSO) [6], [7], which is precisely the approach adopted in the work reported in this paper.

The PSO algorithm was first proposed by J. Kennedy and R. Eberhart in 1995 [8] and it was successfully used in several single-objective optimization problems. PSO is based on the behavior of communities that have both social and individual conducts, similar to birds searching for food.

PSO is a population-based algorithm. Each individual (particle) represents a solution in a $n-$dimensional space. Each particle also has knowledge of its previous best experience and knows the global best experience (solution) found by the entire swarm.

Particles update their exploration directions (their flights) using the following equations:

$$v_{i,j} = w\times v_{i,j}+c_1\times r_1\times(p_{i,j}-x_{i,j})+c_2\times r_2\times(p_{g,j}-x_{i,j}) \quad (1)$$

$$x_{i,j} = x_{i,j} + v_{i,j} \quad (2)$$

where w is the inertia factor influencing the local and global abilities of the algorithm, $v_{i,j}$ is the velocity of the particle i in the $j-th$ dimension, c_1 and c_2 are weights affecting the cognitive and social factors, respectively. r_1 and $r_2 \sim \mathcal{U}(0,1)$; p_i stands for the best value found by particle i (*pbest*) and p_g denotes the global best found by the entire swarm (*gbest*).

After the velocity is updated, the new position i in its $j-th$ dimension is calculated. This process is repeated for every dimension and for all the particles in the swarm.

In order to use PSO for multi-objective optimization problems, our SMOPSO algorithm was hybridized with some concepts taken from de EAs field such as a mutation operator, and with concepts commonly used in MOEAs, such as a selection based on Pareto dominance and mechanisms to produce a good spread of solutions.

The remainder of the paper is organized as follows: Section 2 gives a brief description of the most relevant previous work. Section 3 reviews the basic concepts of multi-objective optimization. Section 4 describes our approach. Section 5 presents the test functions taken from the specialized literature to validate our approach. Section 6 defines the metrics used to evaluate the performance of the algorithms. Section 7 explains the experimental design and gives a short description of PAES and MOGA2, the multi-objective evolutionary algorithms selected for comparing our results. Section 8 shows and discusses the results obtained. Finally, our conclusions and possible future research lines are presented in Section 9.

2. STATE OF ART IN MULTI-OBJECTIVE PSO

In the last few years, several PSO algorithms have been proposed to trackle the multi-objective optimization problem. Here we briefly review the most relevant of them.

In Coello Coello and Lechuga [9], the authors adopt a Pareto-based selection scheme combined with an adaptive grid (similar to the one incorporated in PAES [10]). The adaptive grid is adopted both to store the non-dominated solutions found during the search and to distribute them uniformly along the Pareto frontier.

Hu and Russell [11] proposed their Dynamic Neighborhood PSO, where the neighborhood of each particle is calculated at each iteration, after calculating distances to every other particle. In each new neighborhood the local best particle is identified.

Ray and Lew [12] worked with Pareto dominance and a PSO hybridized with some concepts of EAs. The non-dominated particles are ranked based on Pareto ranking and stored as a set of leaders (SOL). Selection of a leader from the SOL is done with proportional selection to ensure that SOL members with a larger crowding radius have a higher probability of being selected as leaders. The process in turn results in a spread along the Pareto frontier.

Parsopoulos et al. [13] proposed a PSO algorithm using an enhanced elitist technique that consists in maintaining the non-dominated solutions found during the run of the multi-objective algorithm. These solutions are stored in an external archive. A new solution is stored in the file only when it is non-dominated by all the other solutions already stored in archive and it is deleted when it is dominated by some solution stored in the file.

Toscano and Coello Coello [14] proposed the use of clustering techniques to improve the performance of a multi-objective PSO. They used Pareto dominance to guide the flight direction of a particle and had a set of sub-swarms to focalize the search. A PSO algorithm is run in each sub-swarm and at some point the sub-swarms exchange information.

On the other hand, our approach uses a PSO algorithm extended with the concept of Pareto dominance, elitism, a mutation operator and a grid (similar to the one used by PAES) to maintain diversity.

3. BASIC CONCEPTS

The general Multi-Objective Optimization Problem can be defined as follows [9]:
Def. 1: Find the vector $\vec{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ which satisfies the m inequality constrains:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, m \tag{3}$$

and the p equality constrains:

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{4}$$

and optimizes the vector function:

$$\vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T \tag{5}$$

The constrains given by (3) and (4) define the feasible region Ω and any point in Ω defines a feasible solution. The k components of the vector $\vec{f}(\vec{x})$ are the criteria to be considered. The constrains $\vec{g}_i(\vec{x})$ and $\vec{h}_i(\vec{x})$ represent the restrictions imposed on the decision variables. The vector $\vec{x}^*$ denotes the optimum solutions.
When there are several objective functions, the concept of optimum changes, because in multi-objective optimization problems the purpose is to find "trade-off" solutions rather than a single solution. The concept of optimum commonly adopted in multi-objective optimization is the one proposed by Vilfredo Pareto in 1986 (and called Pareto optimality). It is defined as:
Def. 2 *(Pareto Optimality)*: A point $\vec{x}^* \in \Omega$ is Pareto optimal if $\forall \vec{x} \in \Omega$ and $I = \{1, 2, \ldots, k\}$ either:

$$\forall i \in I \left(f_i\left(\vec{x}^*\right) \leq f_i\left(\vec{x}\right)\right) \tag{6}$$

and, there is at least one $i \in I$ such that

$$f_i\left(\vec{x}^*\right) < f_i\left(\vec{x}\right) \tag{7}$$

This definition says that $\vec{x}^*$ is Pareto optimal if there exists no feasible vector $\vec{x}$ which would decrease some criteria without causing a simultaneous increase in at least one other criterion.
Other important definitions associated with Pareto Optimality are the following:
Def. 3 *Pareto Dominance*: A vector $\vec{x} = (x_1, x_2, \ldots, x_k)$ is said to dominate $\vec{y} = (y_1, y_2, \ldots, y_k)$, denoted by $\vec{x} \preceq \vec{y}$, if and

only if $\vec{x}$ is partially less than $\vec{y}$, i.e., $\forall i \in \{1, 2, \ldots, k\}$: $x_i \leq y_i$ and, at least for one i, $x_i < y_i$.

Def. 4 *Pareto Optimal Set*: For a given multi-objective problem $\vec{f}(x)$, the Pareto optimal set, denoted by $\mathcal{P}^*$ or $\mathcal{P}_{true}$, is defined as:

$$\mathcal{P}^* = \{x \in \Omega \mid \nexists x^{'} \in \Omega \vec{f}(x^{'}) \preceq \vec{f}(x)\}. \qquad (8)$$

Def. 5 *Pareto front*: For a given multi-objective problem $\vec{f}(x)$ and Pareto optimal set $\mathcal{P}^*$, the Pareto front, denoted by $\mathcal{PF}^*$ or $\mathcal{PF}_{true}$, is defined as:

$$\mathcal{PF}^* = \{\vec{y} = \vec{f} = (f_1(x), f_2(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\} \quad (9)$$

4. OUR APPROACH

In order to provide a multi-objective approach to PSO we extend the "classical" model described above including:

A Uniform Mutation Operator [15]. It selects one dimension of the particle with a certain probability and changes its value. The new value must be in the range permitted for this dimension.

An elitist policy with the objective of maintaining the best solutions (non-dominated) found in the flight cycles (iterations). The non-dominated solutions are stored in an external archive. This archive has a grid structure (similar to the PAES algorithm) [16] constructed as follows: each objective is divided into 2^d equal divisions. In this way the entire search space is divided into 2^{d^k} unique equal size $k-$dimensional hypercubes (d is a user parameter and k is the number of objective functions). The stored solutions are placed in one of these hypercubes according to their locations in the objective space. The number of solutions in each hypercube is counted. When the archive is full and there is a new non-dominated solution it cannot be included automatically. First, the hypercube that has more non-dominated solutions is found. If the new solution does not belong to that hypercube it is inserted in the archive and at random one of the solutions from the highest covered hypercube is deleted. So, non-dominated solutions are privileged and placed in an archive. When non-dominated solutions compete for a space in the archive, they are evaluated based on how crowded they are in objective function space. The one residing in the least crowded area gets preference. In this manner, we obtain diversity in the non-dominated solutions.

New mechanisms to select the pbest and gbest particles. A multi-objective PSO cannot use equation (1) in a straightforward manner for *pbest* and *gbest* because all non-dominated solutions are equally good. Our approach updates *pbest*, the best experience found for a particle, only when the new particle is non-dominated and it dominates the previous *pbest*. In order to select *gbest*, the global best particle, at each iteration we randomly select a non-dominated particle of the external archive, because by definition all the Pareto optimal solutions are equally good. All other variables in equations (1) and (2) have the meaning defined for the "classic" PSO.

```
1.  SMOPSO{
2.     Init_Pop();
3.     Init_Velocity();
4.     Evaluate_Pop();
5.     Update_Fbest();
6.     Update_Pbest();
7.     Insert_nodom();
8.     Gbestpos = rnd(0,nodomfileSize)
9.     for(i=1 to MAXCYCLES){
10.        for(j=0 to MAXPARTICLES){
11.            Update_Velocity();
12.            Update_Particle();
13.        }
14.        Keeping();
15.        Evaluate_Pop();
16.        Update_Fbest();
17.        Update_Pbest();
18.        Insert_nodom();
19.        Gbestpos = rnd(0,nodomfileSize)
20.    }
21.    Print_Statistics();
22.    Generate_Outfile();
23. }
```

Figure 1: Pseudo-code SMPSO

The pseudo-code of our approach, SMOPSO is shown in Figure 1.

Once all the structures have been allocated (line 1), the particle swarm is initialized with random values corresponding to the ranges of the decision variables, these values are dependent on the test functions. The velocities are initialized with zero values (lines 2-3). Then the swarm is evaluated using the corresponding objective functions (line 4). Next, the fitness vectors are updated (line 5). As we are dealing with multi-objective optimization, these vectors store the values of each decision variable, in which the particles obtained the best values in a Pareto sense. At this stage of the algorithm these vectors are filled with the results of the initial particle evaluations. Analogously, these values are copied in the *pbest* vectors (line 6). Then all non-dominated particles are inserted in the grid, i.e. in the external file (line 7) and the global *gbest* particle is randomly selected (line 8).

The flight cycle starts at line 9, the velocity of each particle is updated, using (1) and its position is also updated using (2) (lines 11-12). The keeping operation is carried out to maintain the particles into the allowable range values (line 13). Then the particles are mutated, evaluated and the fitness and *pbest* vector are, if appropriate, updated (lines 14-17).

As the particles moved in the search space because they changed positions, the dominance of each particle (line 18) is verified and, if appropriate, they are inserted in the grid. Then the new *gbest* is randomly selected (line 19). The cycle is executed until the condition is false and at this point we print the statistics and generate an output file, which contains the non-dominated particles (line 22).

5. TEST FUNCTIONS

In order to validate our approach, we selected the following three well-known test functions [17] :
MOP5: Proposed by Viennet, it is an (unconstrained) three objective function that has its $\mathcal{P}_{true}$ disconnected and asymmetric, and its $\mathcal{PF}_{true}$ is connected. It is defined as:

$$F = (f_1(x,y), f_2(x,y), f_3(x,y)) \text{ with } -30 \leq x, y \leq 30$$
$$f_1(x,y) = 0.5 * (x^2+y^2) + sin(x^2+y^2),$$
$$f_2(x,y) = (3x-2y+4)^2/8 + (x-y+1)^2/27 + 15,$$
$$f_3(x,y) = 1/(x^2+y^2+1) - 1.1e^{(-x^2-y^2)}$$

MOP6: This is a test function constructed using Deb's methodology. It is unconstrained and has two objectives functions. $\mathcal{P}_{true}$ and $\mathcal{PF}_{true}$ are disconnected and its $\mathcal{PF}_{true}$ consists of four Pareto curves. It is defined as:

$$F = (f_1(x,y), f_2(x,y)) \text{ with } 0 \leq x, y \leq 1$$
$$f_1(x,y) = x,$$
$$f_2(x,y) = (1+10y) * [1 - (x/(1+10y))^2 - x/(1+10y) * sin(2\pi 4x)]$$

MOPC1: This is a function with constraints and two objectives and it was proposed by Binh and Korn. In this case $\mathcal{P}_{true}$ is an area and $\mathcal{PF}_{true}$ is convex. It is defined as:

$$F = (f_1(x,y), f_2(x,y)) \text{ with } 0 \leq x \leq 5, \; 0 \leq y \leq 3 \text{ and}$$
$$f_1(x,y) = 4x^2 + 4y^2,$$
$$f_2(x,y) = (x-5)^2 + (y-5)^2$$

subject to:

$$0 \geq (x-5)^2 + y^2 - 25$$
$$0 \geq -(x-8)^2 - (y+3)^2 + 7.7$$

6. EXPERIMENTAL PERFORMANCE METRICS

Usually among the relevant aspects to measure in the performance of a multi-objective optimization algorithm, there are two that are very important: 1) the spread across the Pareto optimal front and 2) the ability to attain the final trade-off solutions.
In this direction, we select the following metrics to evaluate the performance of our approach:

Generational Distance (GD): proposed by [18] this metric returns a value representing the average distance of the solutions in the Pareto front constructed by a multi-objective algorithm ($\mathcal{PF}_{known}$) from $\mathcal{PF}_{true}$ and it is defined as:

$$\boldsymbol{GD} = \frac{1}{n}\sqrt{\sum_{i=1}^{n} d_i^2}$$

where n is the number of solutions in $\mathcal{PF}_{known}$ and d_i is the Euclidean distance (in objective space) between each vector in $\mathcal{PF}_{known}$ and the nearest member of $\mathcal{PF}_{true}$. A zero result indicates that both fronts are the same, any other value indicates $\mathcal{PF}_{known}$ deviates from $\mathcal{PF}_{true}$.
Spacing (S): [19] proposed a metric which allows to measure the distribution of vectors throughout $\mathcal{PF}_{known}$. It is defined as:

$$\boldsymbol{S} = \sqrt{\frac{1}{(n-1)}\sum_{i=1}^{n}(\overline{d} - d_i)^2}$$

where n is the number of solutions in $\mathcal{PF}_{known}$, $d_i = min_j(\mid f_1^i(\vec{x}) - f_1^j(\vec{x}) \mid + \mid f_2^i(\vec{x}) - f_2^j(\vec{x}) \mid)$, $i, j = 1, \ldots, n$ and $\overline{d}$ is the mean of all d_i.
A zero value for this metric means that all members of $\mathcal{PF}_{known}$ are equidistantly spaced.

7. EXPERIMENTS

The experiments were designed to evaluate the performance of SMOPSO to existing models on the three test functions described in section 5.

The first comparative model is PAES algorithm [10], which is a $(1+1)$-ES evolution strategy (i.e. it applies the mutation operator over one individual and generates only one child). It also implements elitism using an external file to store the non-dominated individuals. This algorithm was selected because it has the same grid technique that we implement in SMOPSO for maintaining diversity.

The second model is MOGA2 [20], a genetic algorithm that uses a Pareto ranking technique, in which the rank of an individual x_i is calculated as $x_i = 1 + n_i$, where n_i is a penalty and indicates the number of individuals that dominate x_i, so all the non-dominated individuals have a rank 1. Also, to avoid premature convergence and to maintain diversity among the non-dominated solutions, it implements a niching technique. The original MOGA was extended with an elitist policy and this last version, MOGA2, is the one used in this work.

The initial parameter settings for all the algorithms are summarized in Table 1. The best algorithm parameter values were empirically derived

Table 1: Parameter Settings

Parameters	SMPSO		
	MOP5	*MOP6*	*MOPC1*
Iterations	7000	3000	2000
Extern file size	799	799	799
Crossover prob.	-	-	-
Mutation prob.	0.5	0.0335	0.3
Particles/Individuals	30	20	20
Divisions	5	5	5
C1=C2	1.5	1.6	1.5
W	0.5	0.6	0.5

Parameters	PAES		
	MOP5	*MOP6*	*MOPC1*
Iterations	210000	60000	40000
Extern file size	799	799	799
Crossover prob.	-	-	-
Mutation prob.	0.03	0.05	0.05
Particles/Individuals	1	1	1
Divisions	5	6	5

Parameters	MOGA2		
	MOP5	*MOP6*	*MOPC1*
Iterations	7000	3000	2000
Extern file size	799	799	799
Crossover prob.	0.8	0.8	0.8
Mutation prob.	0.025	0.025	0.025
Particles/Individuals	30	20	20

Table 2: Values of the Performance Metrics

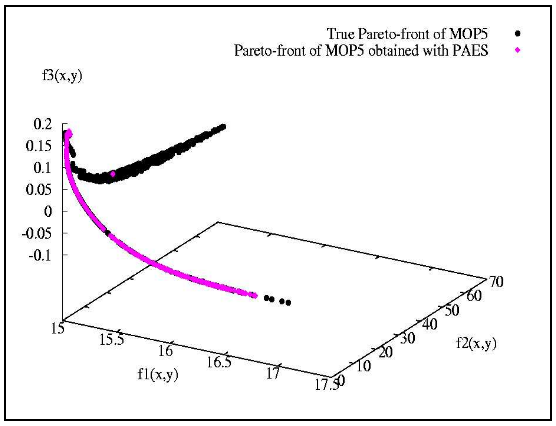
Metric	GD		
	SMOPSO	*PAES*	*MOGA2*
MOP5	0.011083	0.167133	0.727717
MOP6	0.000298	0.02204	0.000999
MOPC1	0.002687	0.01986	2.441103

Metric	S		
	SMOPSO	*PAES*	*MOGA2*
MOP5	0.39566	0.378168	0.200790
MOP6	0.003402	0.011853	0.009080
MOPC1	0.116149	0.32289	0.942198

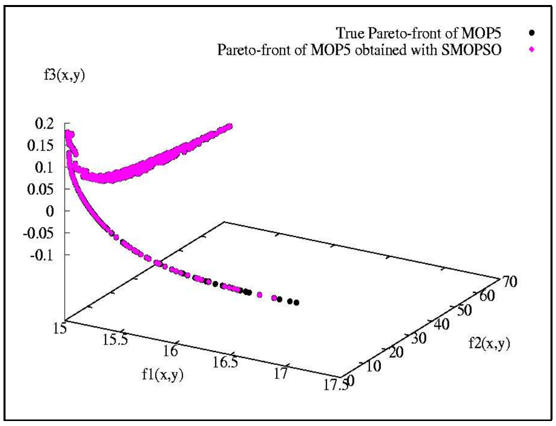
from a set of previous experiments. We ran the algorithms up to a maximum number of evaluations ($iterations \times individuals$) of the multi-objective function, in order to compare them on the basis of the same amount of computational effort. The entry Divisions, in Table 1, indicates the number of hypercubes in the grid used to maintain diversity. Each algorithm was executed ten times on each test function.

8. RESULTS AND DISCUSSION

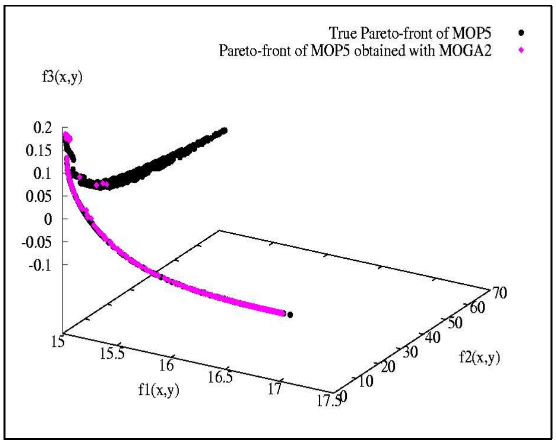
The solutions obtained for the algorithms (PF_{know}) were compared with the true Pareto fronts (PF_{true}) for each function. These last were obtained using an enumerative method with the following accuracy of the decision variables: 0.05 for MOP5, 0.003 for MOP6, and 0.01 for MOPC1.
The values shown in Table 2 correspond to mean values calculated over the ten runs performed in each experiment. The performance of SMOPSO across the benchmark functions is comparable or overcomes the performance of PAES and MOGA2. SMOPSO obtained better metric values that the other algorithms. This indicates that the solutions in PF_{know} are the same or are very close to the PF_{true} with a good distribution of points in all the cases.



(a) MOP5 with PAES.



(b) MOP5 with SMOPSO.



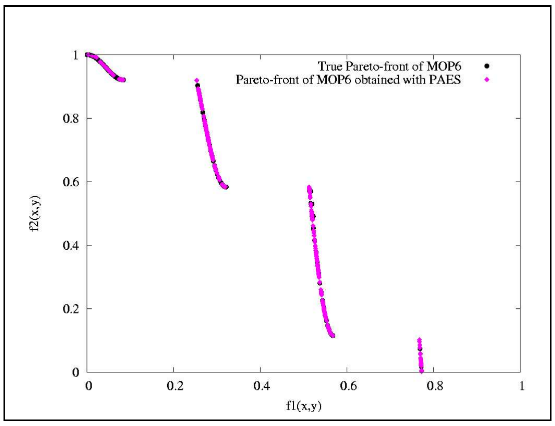
(c) MOP5 with MOGA2.

Figure 2: MOP5 with PAES, SMOPSO and MOGA2

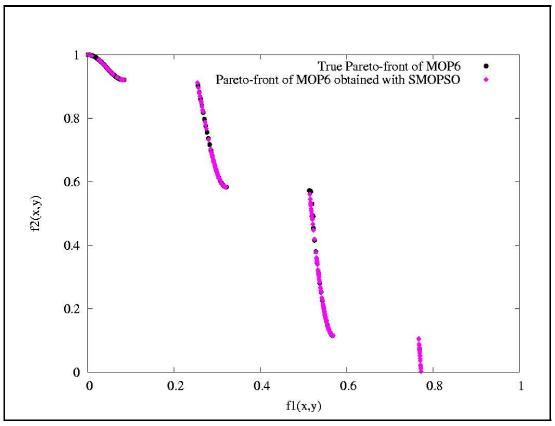
However, in the graphics of PF_{know} and PF_{true} shown in Figure 2, it can be observed that even though MOGA2 obtained a better distribution of solutions (better value of S metric), for the MOP5 function, it can not approximate the top portion of the Pareto front.

The same occurs with the solutions found by PAES. Only SMOPSO approximated solutions over both curves of the front.

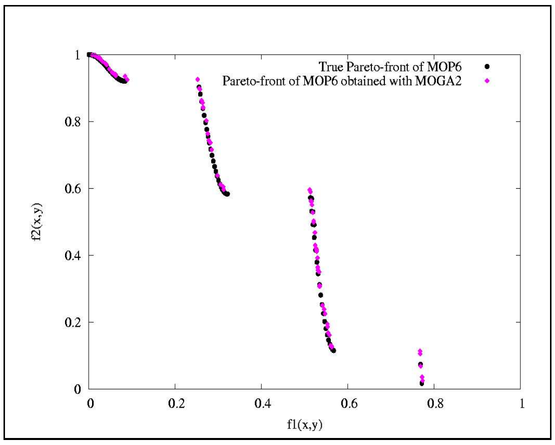
The function with disconnected front MOP6 (Figure 3) was not difficult for any algorithm, because all of them obtained PF_{know} very close to PF_{true} with a good distribution of solutions.



(a) MOP6 with PAES.

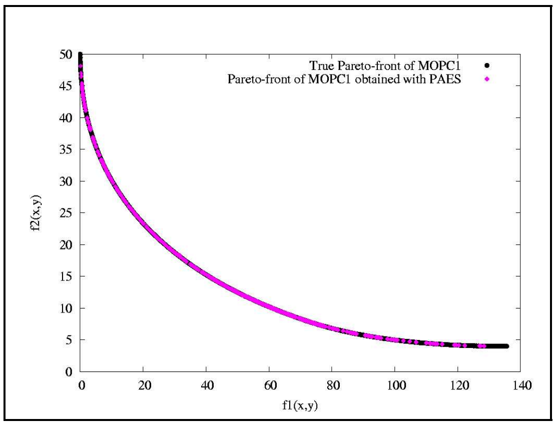


(b) MOP6 with SMOPSO.

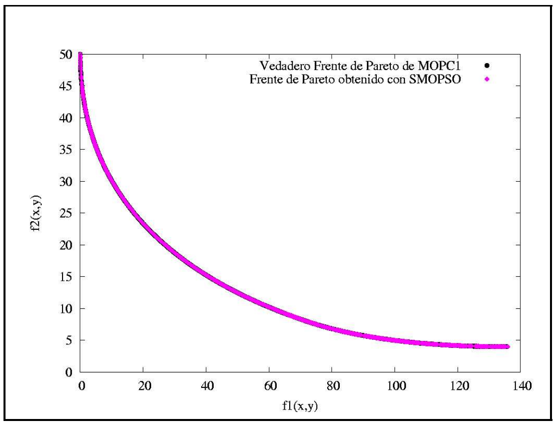


(c) MOP6 with MOGA2.

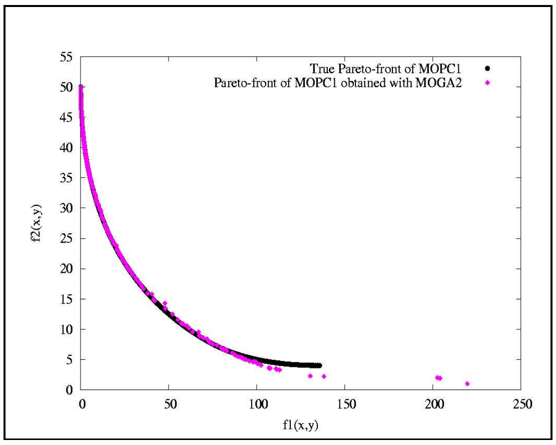
Figure 3: MOP6 with PAES, SMOPSO and MOGA2



(a) MOPC1 with PAES.



(b) MOPC1 with SMOPSO.



(c) MOPC1 with MOGA2.

Figure 4: MOPC1 with PAES, SMOPSO and MOGA2

Regarding the function with constraints, MOPC1 (Figure 4), although the three algorithms found good approximations to the true Pareto front we observe that the distance among solutions generated by PAES, in the bottom of the front, are greater than the solutions generated by SMOPSO. Furthermore MOGA2 produces solutions at the bottom which are away from the true Pareto front.

PAES and MOGA2 showed weaknesses at finding some parts of the true Pareto fronts, whereas SMOPSO does have not this problem. This fact is perhaps due to the strong exploration of the search space provided by the mutation operator at every flight cycle.

9. CONCLUSIONS AND FUTURE WORK

The performance of our approach over all the benchmark functions studied turned out to be satisfactory in the sense that the aim was to determine if the new algorithm was able to obtain results at least comparable to those obtained by two well-known MOEAs.

The results showed that, in spite of its simplicity, the SMOPSO is a promising approach to multi-objective optimization because its performance was generally better that the models used for comparison. As part of our ongoing work we are going to compare SMOPSO with other more competitive MOGAs as NSGA-II [21], [22] and the Micro-Genetic Algorithm [23].

References

[1] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsgaii. In *Parallel Problem Solving from Nature, PPSN VI*, pages 849–858. Springer, 2000.

[2] J. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *First International Conference on Genetic Algorithms*, pages 99–100, 1985.

[3] E. Zitzler, K. Deb, and L. Thiele. Comparision of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

[4] J. Knowles and D. Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[5] M. Laumanns, E. Zitzler, and L. Thiele. A unified model for multi-objective evolutionary algorithms with elitism. In *Congress on Evolutionary Computation*, pages 46–53, Piscataway, NJ, 2000. IEEE Service Center.

[6] J. Kennedy and R. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, California, USA, 2001.

[7] J. Kennedy and R. Mendes. Population structure and particle swarm performance. In *Congress on Evolutionary Computation*, volume 2, pages 1671–1676, Piscataway, NJ, May 2002. IEEE Sevice Center.

[8] J. Kennedy and R. Eberhart. Particle swarm optimization. In *International Conference on Neural Networks*, pages 1942–1948, Piscataway, NJ., 1995. IEEE Sevice Center.

[9] C. Coello Coello and M. Lechuga. Mopso: A proposal for multiple objective particle swarm optimization. In *Congress on Evolutionary Computation*, pages 1051–1056, Piscataway, NJ., 2002. IEEE Service Center.

[10] J. Knowles and D. Corne. M-paes: A memetic algorithm for multiobjective optimization. In *Congress on Evolutionary Computation*, pages 325–332, Piscataway, NJ, 2000. IEEE Service Center.

[11] X. Hu and R. Eberhart. Multiobjective optimization using dynamic neigborhood particle swarm optimization. In *Congress on Evolutionary Computation*, pages 1677–1681, Piscataway, NJ., 2002. IEEE Service Center.

[12] T. Ray and K. Liew. A swarm metaphor for multiobjective design optimization. *Engineering Optimization*, 34(2):141–153, 2002.

[13] K. Parsopoulos, T. Bartz, and Vrahatis. Particle swarm optimizers for pareto optimization with enhanced archiving techniques. In *Congress on Evolutionary Computation*, pages 1780–1787, Piscatawy, NJ., 2003. IEEE Service Center.

[14] C. Coello Coello and G. Toscano Pulido. Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer. In *Genetic and Evolutionary Computation Conference*, volume 3102 of *Lecture Notes in Computer Science*. Springer Verlag, 2004.

[15] T. Bäck, D. Fogel, and Z. Michalewicz, editors. *Handandbook of Evolutionary Computation*. IOP Publishing Ltd and Oxford University Press, 1997.

[16] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Ltd., England, 2001.

[17] C. Coello Coello, D. Van Vedhuizen, and G. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, USA, 2002.

[18] D. Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Air Force Institute of Technology, Dayton, 1999.

[19] J. Schott. Fault tolerant design using single and multi-criteria genetic algorithms. Master's thesis, Massachusetts Institute of Technology, Boston, 1995.

[20] C. Fonseca and P. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. Morgan Kaufmann Publishers.

[21] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: Nsga ii. Technical Report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[22] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: Nsga ii. In Schoenauer M. et al., editor, *Parallel Problem Solving from Nature*, Lecture Notes in Computer Sciences 1917, pages 849–858. Springer, 2000.

[23] C. Coello Coello and G. Toscano Pulido. Multiobjective optimization using a micro genetic algorithm. In Goodman L. et al., editor, *Genetic and Evolutionary Conference*, pages 274–282, San Francisco, California, 2001. Morgan Kaufmann Publishers.