A PROPOSAL TO USE STRIPES TO MAINTAIN DIVERSITY IN A MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZER

Mario Alberto Villalobos-Arias[†], *Gregorio Toscano Pulido*[‡] *and Carlos A. Coello Coello*[‡]

ABSTRACT

In this paper, we propose a new mechanism to maintain diversity in multi-objective optimization problems. The proposed mechanism is based on the use of stripes that are applied on objective function space and that is independent of the search engine adopted to solve the multi-objective optimization problem. In order to validate the proposed approach, we included it in a multi-objective particle swarm optimizer. Our approach was compared with respect to two multi-objective evolutionary algorithms which are representative of the state-of-the-art in the area. The results obtained indicate that our proposed mechanism is a viable alternative to maintain diversity in the context of multi-objective optimization.

1. INTRODUCTION

In the last few years, several multi-objective particle swarm optimizers (MOPSOs) have been proposed in the specialized literature (see for example [2, 12, 14, 5, 7, 8, 15, 13]).

Most of this work, however, focuses mainly on the design of novel selection or archiving mechanisms. Nevertheless, the design of effective mechanisms to maintain diversity remains as a key issue when extending particle swarm optimizers so that they can deal with multi-objective optimization problems.

In some recent work, a few authors have proposed or adopted novel mechanisms to maintain diversity in their MOPSOs (e.g., [11, 10, 15]). Such approaches have led to the development of very successful multi-objective particle swarm optimizers.

In this paper, we propose a new mechanism to maintain diversity, which we show to overcome the main drawbacks of other popular mechanisms such as ε-dominance [6] and the sigma method proposed in [11].

The remainder of the paper is organized as follows. Section 2 presents some basic concepts related to multi-objective optimization in general. Section 3 presents the most relevant previous related work. Our proposed approach is described in Section 4. Section 5 presents a comparison of the results produced by our approach (coupled to a multi-objective particle swarm optimizer) and two multi-objective evolutionary algorithms that are representative of the state-of-the-art. Finally, in Section 6, we present our conclusions and some possible paths for future research.

[†]CINVESTAV-IPN, Departamento de Matemáticas, Av. IPN No. 2508, Col. San Pedro Zacatenco, México, D.F. 07360, MEXICO, mava@math.cinvestav.mx

[‡]CINVESTAV-IPN, Evolutionary Computation Group, Sección Computación, Av. IPN No. 2508, Col. San Pedro Zacatenco, México, D.F. 07360, MEXICO, gtoscano@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx

2. BASIC CONCEPTS

The multi-objective optimization problem (MOP) we are concerned with is to find a vector $\vec{x}^* \in X \subset I\!R^m$ such that

$$F(\vec{x}^*) = \min_{\vec{x} \in X} F(\vec{x}) = \min_{\vec{x} \in X} [f_1(\vec{x}), \ldots, f_d(\vec{x})], \quad (1)$$

where $F : X \subset I\!R^m \longrightarrow I\!R^d$ is a given vector function with components $f_i : X \subset I\!R^m \longrightarrow I\!R$ for each $i \in \{1, \ldots, d\}$, and the minimum is understood in the sense of the standard Pareto order on $I\!R^d$, which is defined as follows.

If $\vec{u} = (u_1, u_2, \ldots, u_d)$ and $\vec{v} = (v_1, v_2, \ldots, v_d)$ are vectors in $I\!R^d$, then $\vec{u} \preceq \vec{v} \iff u_i \leq v_i \; \forall \; i \in 1, \ldots, d$. This relation is a partial order.
We also have $\vec{u} \prec \vec{v} \iff \vec{u} \preceq \vec{v}$ and $\vec{u} \neq \vec{v}$.

Definition 1 *A vector $\vec{x}^* \in X$ is called a* Pareto optimal solution *for the multi-objective optimization problem (MOP) if there is no $\vec{x} \in X$ such that $F(\vec{x}) \prec F(\vec{x}^*)$.*

The set $\mathcal{P}^* = \{\vec{x} \in X \; : \; \vec{x}$ is a Pareto optimal solution$\}$ is called the Pareto optimal set *and its image under F, i.e. $F(\mathcal{P}^*) := \{F(\vec{x}) \; : \; \vec{x} \in \mathcal{P}^*\}$, is the* Pareto front.

Lemma 1 *Let $\vec{x}^1, \vec{x}^2, \ldots, \vec{x}^d \in X$ be the minimals of the functions $f_1, f_2, \ldots, f_d$. Then the Pareto front is contained in the "Hyper-box" defined by the points $F(\vec{x}^1), F(\vec{x}^2), \ldots, F(\vec{x}^d)$.*

The proof of Lemma 1 is trivial and is, therefore, omitted here. The Lemma 1 is illustrated in Figure 1, for the case in which $d = 2$ and the Pareto front corresponds to the parts on the boundary of S joining the points A and B, and also the points C and D,

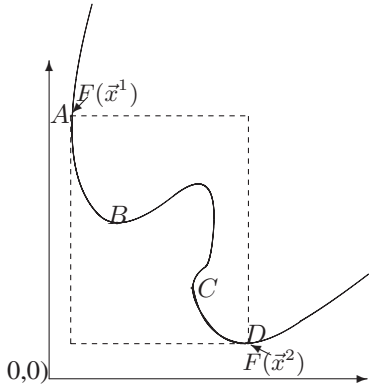

Figure 1. $F(\mathcal{P}^*)$ is contained in the "Hyper-box" defined by $F(\vec{x}^1),\ F(\vec{x}^2)$

We present the well–known "scalarization" result that we will use later on.

Lemma 2 *Let $\vec{x}^* \in X$ be a solution of the weighted problem:*

$$\min_{\vec{x} \in X} \sum_{s=1}^{n} w_s f_s(\vec{x}),$$

where $w_s \geq 0\ \ \forall s \in \{1, \ldots, n\}$ and $\sum_{s=1}^{n} w_s = 1$.
Then $\vec{x}^* \in \mathcal{P}^*$.

Proof See, for instance, [9, p.78].

3. PREVIOUS RELATED WORK

There are two main approaches to maintain diversity of MOPSOs that have been reported in the specialized literature: the sigma method proposed by Mostaghim et al. [11] and the ε-dominance method proposed by Laumanns et al. [6].

The sigma method uses the vector that the evaluation of $F(\vec{x})$ of the particle $\vec{x}$ represents, and the leader of this particle is the individual in the elite set whose sigma is closest to the sigma of $\vec{x}$ (sigma is a direction and is computed using an expression provided by the authors of this method [11]). The core idea in the sigma method is to form clusters using the particles in the elite set as the centers of such clusters. Note however that the elite set could be very large. Since the number of elements in each cluster is not bounded, there could be leaders with many "followers" and some leaders with no "followers". In consequence, the approach may fail to cover all the Pareto front. Also, the approach requires that all the objective function values are positive (some sort of scaling is required when this is not the case). Figure 2 shows a case in which the sigma method could fail. In this figure, all the directions go to the portion of the Pareto front which is closer to the "ideal vector". Thus, it is possible that the solutions generated do not cover all the Pareto front.

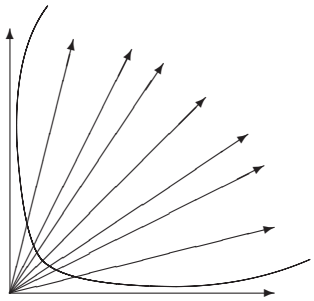

Figure 2. This figure illustrates a situation that causes problems to the sigma method proposed by Mostaghim et al. [11].

The concept of ε-dominance [6] refers to a relaxed form of dominance. A decision vector x_1 is said to ε-dominate a decision vector x_2 for some $\varepsilon > 0$ iff: $f_i(x_1)/(1+\varepsilon) \leq f_i(x_2), \forall i = 1, ..., m$ and $f_i(x_1)/(1+\varepsilon) < f_i(x_2)$, for at least one $i = 1, ..., m$ (m is the total number of objective functions of the problem). It is worth noting that ε is a user-defined parameter.

This concept is normally used to fix the size of the external archive (or secondary population) in which a multi-objective evolutionary algorithm retains the nondominated vectors found during the search.

The main drawback of the $\varepsilon-$dominance method is the number of comparisons and distances that have to be computed. Another possible problem with the ε-dominance approach is shown in Figure 3. In this case, the point A is closer to the lower lefthand corner than point B, but point B is closer to the Pareto front than point A. So, in this case, the ε-dominance approach retains point A. In contrast, our approach will retain point B.

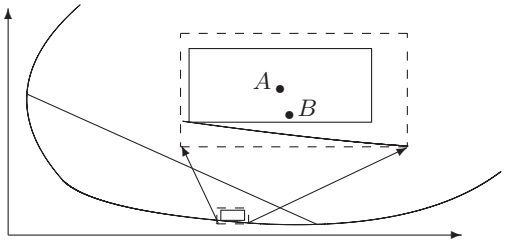

Figure 3. An example in which the ε-dominance approach retains the wrong point.

4. OUR PROPOSAL

In the rest of the paper we will assume that $d = 2$, but the proposed approach can be generalized to any number of di-

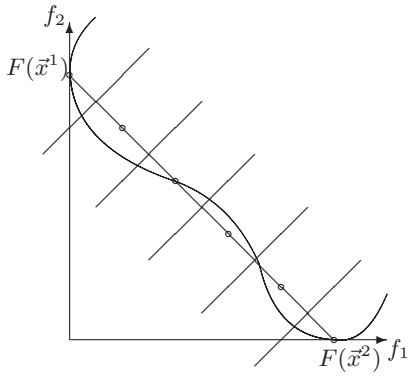

Figure 4. Graphical representation of the stripes proposed in this paper.

mensions. The core idea of the approach proposed in this paper (which we call "stripes") is that the line generated by the points $F(\vec{x}^1), F(\vec{x}^2)$ (defined in Lemma 1) is "similar" to the Pareto front. Thus, we can use several points (which we call stripe centers) uniformly distributed along this line, and we assign the individuals of the population to the nearest stripe center. This way, we are distributing the individuals in several stripes determinated by the stripe centers (see Figure 4). Now, if we set an upper bound on the number of individuals in each stripe and on the number of elements of the Pareto front, the approach will provide a distribution of points, avoiding an excessive clustering in any particular region from those defined by the stripes. In this paper, we use the notion of clustering, but the center of each cluster is fixed and uniformly distributed along a line, as shown in Figure 4 (the small circles are the centers of the clusters).

The stripe center set can be computed using

$$X_i = \frac{iF(x^1) + (ns-1-i)F(x^2)}{ns-1}, \quad i \in \{0, 1, \ldots, ns-1\}, \tag{2}$$

where ns is the number of stripes, which is a parameter provided by the user.

In the case in which there are only two objective functions, $d = 2$, we can apply a rotation to all elements in the population and to all elements in the elite set, such that the vector $F(\vec{x}^1) - F(\vec{x}^2)$ is parallel to the $x-$axis. Then, the stripe of every element in the population is calculated using the coordinate x of the rotated element, as follows. Let θ be the angle between the $x-$axis and the vector $F(\vec{x}^1) - F(\vec{x}^2)$. Thus, this angle is what we need to rotate all the elements. Then, if $F^r(\vec{x}) = (f_1^r(\vec{x}), f_2^r(\vec{x}))$ are the rotated coordinates of $F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}))$, we have

$$\begin{aligned} f_1^r(\vec{x}) &= \cos(\theta) f_1(\vec{x}) - \sin(\theta) f_2(\vec{x}) \\ f_2^r(\vec{x}) &= \sin(\theta) f_1(\vec{x}) + \cos(\theta) f_2(\vec{x}) \end{aligned} \tag{3}$$

Now, to determine the stripe of the individual whose evaluation is $F(\vec{x})$ we use the following expressions. Let

$$h = \frac{f_1^r(\vec{x}^2) - f_1^r(\vec{x}^1)}{ns-1}, \quad \text{and } h_{\vec{x}} = \frac{f_1^r(\vec{x}) - f_1^r(\vec{x}^1)}{h}$$

then

$$\text{stripe}(\vec{x}) = \begin{cases} 1 & \text{if} \quad h_{\vec{x}} < 0.5 \\ [\![h_{\vec{x}} + 1.5]\!] & \text{if} \quad 0.5 \leq h_{\vec{x}} < ns - 0.5 \\ ns & \text{if} \quad h_{\vec{x}} \geq ns - 0.5 \end{cases}$$

To illustrate the way in which our proposed approach works, we show in Figure 5 an example for a problem with two objectives. In the figure, it can be seen that the approach (which was coupled to a multi-objective particle swarm optimizer previously proposed by us [2]).

4.1. PSO with stripes

In order to validate the effectiveness of our proposed approach to maintain diversity, we used a multi-objective particle swarm optimizer previously proposed by us [2] as our search engine. However, in this case, the diversity maintenance scheme are the stripes proposed in this paper instead of the adaptive grid originally adopted [2]. We call our MOPSO with stripes ST-MOPSO.

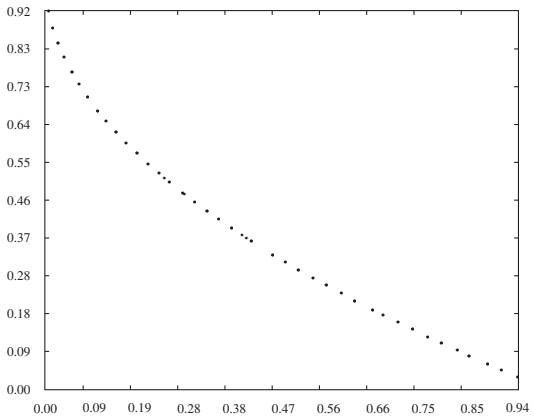

Figure 5. An example of the type of distribution of non-dominated solutions produced by our proposed approach.

Our proposal consists of using one leader in each stripe and to compute a weighted sum determined by the points $F(\vec{x}^1)$ and $F(\vec{x}^2)$, to select the leaders (i.e., the leader of a stripe is the point that minimizes this weighted sum). However, since in our case we have the rotated coordinates $F^r(\vec{x})$, then the leader of a stripe is the particle in the elite set that minimizes f_2^r (the y-coordinate of F^r).

5. COMPARISON OF RESULTS

Several test functions were taken from the specialized literature to validate our approach. Due to space restrictions, we will include results only for three of these test functions. Our results are compared with respect to those produced by two multi-objective evolutionary algorithms representative of the state-of-the-art in the area: the NSGA-II [4] and ε-MOEA [3].

In the results shown next, each approach performed 3000 fitness function evaluations. The results shown correspond to 30 independent runs. In order to allow a quantitative comparison of results we adopted the following metrics: two set coverage [19, 18], Hypervolume [18], inverted generational distance [16], and success counting (which is a variation of the metric called "error ratio" [16]).

The definition of each of these metrics is presented next.

Definition 2 (Two Set Coverage (TSC)) *: This metric can be termed relative coverage comparison of two sets. Let $\wp(X)$ be the power set of X, then TSC is defined as follows:*

$$TSC : \wp(X) \times \wp(X) \longrightarrow [0, 1],$$

$$SC(X', X'') := \frac{|\{x'' \in X''; \exists\, x' \in X' : F(x') \succeq F(x'')\}|}{|X''|}$$

$\forall\, X', X'' \subseteq X$.

If all points in X' dominate or are equal to all points in X'', then by definition $TSC = 1$. $SC = 0$ implies that none of the points in X'' are dominated by X'. In general, $TSC(X', X'')$ and $TSC(X'', X')$ both have to be considered due to set intersections not being empty.

Definition 3 (Hypervolume (HV)) *This metric was proposed by Zitzler and Thiele [18]. The hypervolume defines the area (for the two dimensional case) of objective value space covered by the solution of an algorithm ($\mathcal{P}_a^*$) (i.e., the "area under the curve"). For example, a vector in $\mathcal{P}_a^*$ for a two-objective multi-objective problem defines a rectangle bounded by an origin and $(f_1(\vec{x}); f_2(\vec{x}))$. The union of all such rectangles' area defined by each vector in $\mathcal{P}_a^*$ is then the comparative measure and is defined as:*

$$HV = \left\{ \bigcup_i a_i \mid x_i \in \mathcal{P}_a^* \right\},$$

where x_i is a nondominated vector in $\mathcal{P}_a^*$ and a_i is the Hypervolume determined by the components of x_i and the origin.

Definition 4 (Inverted Generational Distance (IGD)) *The concept of generational distance was introduced by Van Veldhuizen & Lamont [16, 17] as a way of estimating how far are the elements in the Pareto front produced by our algorithm from those in the true Pareto front of the problem. This measure is defined as:*

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{4}$$

where n is the number of nondominated vectors found by the algorithm being analyzed and d_i is the Euclidean distance (measured in objective space) between each of these and the nearest member of the true Pareto front. It should be clear that a value of $GD = 0$ indicates that all the elements generated are in the true Pareto front of the problem. Therefore, any other value will indicate how "far" we are from the global Pareto front of our problem. In our case, we implemented an "inverted" generational distance measure (IGD) in which we use as a reference the true Pareto front, and we compare each of its elements with respect to the front produced by an algorithm. In this way, we are calculating how far are the elements of the true Pareto front, from those in the Pareto front produced by our algorithm. Computing this "inverted" generational distance value reduces the bias that can arise when an algorithm didn't fully cover the true Pareto front.

Definition 5 (Success Counting (SC)) *We define this measure based on the idea of the measure called* Error Ratio *proposed by Van Veldhuizen [16] which indicates the percentage of solutions (from the nondominated vectors found so far) that are not members of the true Pareto optimal set. In this case, we count the number of vectors (in the current set of nondominated vectors available) that are members of the Pareto optimal set:*

$$SC = \sum_{i=1}^{n} s_i,$$

where n is the number of vectors in the current set of nondominated vectors available; s_i = 1 if vector i is a member of the Pareto optimal set, and $s_i = 0$ otherwise. It should then be clear that $SC = n$ indicates an ideal behavior, since it would mean that all the vectors generated by our algorithm belong to the true Pareto optimal set of the problem. For a fair comparison, when using this measure, all the algorithms should limit their final number of nondominated solutions to the same value. Note that SC avoids the bias introduced by the Error Ratio measure, which normalizes the number of solutions found (which belong to the true Pareto front) and, therefore, provides only a percentage of solutions that reached the true Pareto front. This percentage does not provide any idea regarding the actual number of non-dominated solutions that each algorithm produced.

5.1. ZDT1's test function

$$
\begin{aligned}
\text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\
f_1(\vec{x}) &= x_1 \\
f_2(\vec{x}) &= g(\vec{x})\, h(f_1(\vec{x}), g(\vec{x})) \\
g(\vec{x}) &= 1 + 9 \sum_{i=2}^{m} \frac{x_i}{(m-1)}, \\
h(x, y) &= 1 - \sqrt{\frac{x}{y}}
\end{aligned}
$$

where $m = 30$, and $x_i \in [0, 1]$.

TSC	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0.999333	0.999666
epsMOEA	0		0.411102
NSGA-II	0	0.0983892	

HV	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0	0
epsMOEA	0.0203594		0.000509322
NSGA-II	0.0303835	0.0107699	

Table 1. Results of the Two Set Coverage and Hyper Volume metrics for the ZDT1's test function.

IGD	ST-MOPSO	ε-MOEA	NSGA-II
Best	**0.000343805**	0.00167212	0.0020484
Worst	**0.000670735**	0.0190439	0.0276651
Mean	**0.000430186**	0.00795849	0.00641032
Stdev	**7.39891e-05**	0.0050593	0.0052243
Median	**0.000419506**	0.00655261	0.00495141

SC	ST-MOPSO	ε-MOEA	NSGA-II
Best	100	2	8
Worst	95	0	0
Mean	99.3	0.3	1.1
Stdev	1.20773	0.534983	1.66816
Median	100	0	1

Table 2. Results of the Inversed Generational Distance and Sucess Counting metrics for the ZDT1's test function.

Figure 6 shows the graphical results produced by ST-MOPSO, ε-MOEA and the NSGA-II in the first test function chosen. (The true Pareto front of the problem is shown as a continuous line in the left handside picture). Tables 1 and 2 show the comparison of results among the three algorithms considering the metrics previously described. It can be seen that the performance of ST-MOPSO is the best with respect to all the metrics tested. By looking at the Pareto fronts produced by each algorithm in this test function, it should be clear that ST-MOPSO was the only algorithm that could reach the true Pareto front in most of the runs performed (the output of the 30 independent runs was combined in a single file in order to generate the plots from Figure 6).

5.2. ZDT2's test function

$$
\begin{aligned}
\text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\
f_1(\vec{x}) &= x_1 \\
f_2(\vec{x}) &= g(\vec{x})\, h(f_1(\vec{x}), g(\vec{x})) \\
g(\vec{x}) &= 1 + 9 \sum_{i=2}^{m} \frac{x_i}{(m-1)}, \\
h(x, y) &= 1 - \left(\frac{x}{y}\right)^2
\end{aligned}
$$

where $m = 30$, and $x_i \in$[0,1].

Figure 7 shows the graphical results produced by ST-MOPSO, the NSGA-II [4], and ε-MOEA [3] in the second test function adopted.

TSC	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0.993918	0.993918
epsMOEA	0		0.485246
NSGA-II	0	0.012	

HV	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0	0
epsMOEA	0.0377096		0.000175564
NSGA-II	0.0610739	0.0235162	

Table 3. Results of the Two Set Coverage and Hypervolume metrics for the ZDT2's test function.

Tables 3 and 4 show the comparison of results among the three algorithms considering the metrics previously indicated. As in the previous example, the performance of ST-MOPSO was the best with respect to all the metrics adopted.

Graphically (see Figure 7), it can be seen that in this case, our ST-MOPSO generated a few points outside the true Pareto front in one of the runs. However, when looking at the graphical output generated by the other algorithms, it is clear that our ST-MOPSO had the most robust behavior in this problem, since the others produced a considerably large number of solutions outside the true Pareto front of the problem.

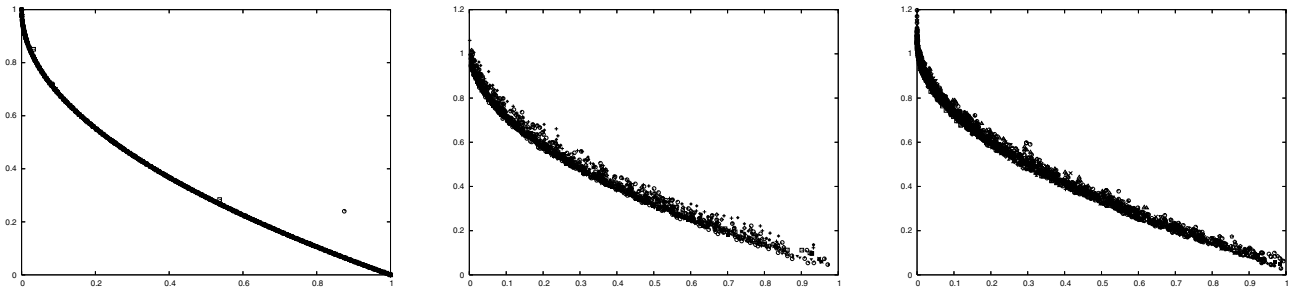

Figure 6. Pareto fronts produced by ST-MOPSO (left), ε-MOEA (center) and NSGA-II (right) for the ZDT1's test function.

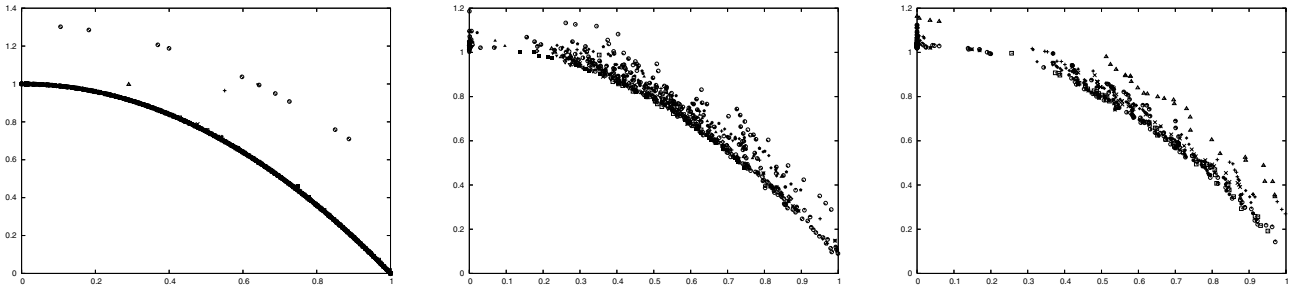

Figure 7. Pareto fronts produced by our ST-MOPSO (left), ε-MOEA (center) and NSGA II (right) for the test ZDT2's function.

IGD	ST-MOPSO	ε-MOEA	NSGA-II
Best	**0.000346726**	0.0038222	0.00461677
Worst	**0.051142**	0.0516044	0.0550343
Mean	**0.0104091**	0.016797	0.0373733
Stdev	**0.0186685**	0.0128839	0.0202878
Median	**0.000439715**	0.0112029	0.0518229

SC	ST-MOPSO	ε-MOEA	NSGA-II
Best	**100**	0	0
Worst	**1**	0	0
Mean	**75.6**	0	0
Stdev	41.8384	**0**	**0**
Median	**100**	0	0

Table 4. Results of the Inversed Generational Distance and Sucess Counting metrics for the ZDT2's test function.

5.3. ZDT3's test function

$$
\begin{aligned}
\text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\
f_1(\vec{x}) &= x_1 \\
f_2(\vec{x}) &= g(\vec{x})\, h(f_1, g) \\
g(\vec{x}) &= 1 + 9 \sum_{i=2}^{m} \frac{x_i}{(m-1)}, \\
h(x, y) &= 1 - \sqrt{\frac{x}{y}} - \frac{x}{y} \sin(10\pi x)
\end{aligned}
$$

where $m = 30$, and $x_i \in$[0,1].

Figure 8 shows the graphical results produced by the ST-MOPSO, the NSGA-II, and ε-MOEA in the third test function chosen.

Tables 5 and 6 show the comparison of results among the three algorithms considering the metrics previously described.

Once more, our ST-MOPSO had the best performance with respect to all the metrics considered. Graphically (see Figure 8), it can be seen that in this case, our ST-MOPSO generated some points outside the true Pareto front in some of the runs. However, when looking at the graphical output generated by the other algorithms, it is clear that our ST-MOPSO had the most robust behavior in this problem, since the others produced a considerably large number of solutions outside the true Pareto front of the problem (this is corroborated by the values of the metrics).

Summarizing our results, it can be seen that the performance of our ST-MOPSO is the best with respect to all the metrics tested. By looking at the Pareto fronts of the three test functions adopted, it can be easily seen that most of the executions of the ST-MOPSO algorithm reached the true Pareto front, which is an indicative of the robustness of the approach. This contrasts with the other approaches, which not only showed a higher variation of results, but were also unable to reach the true Pareto front in most of the runs (this

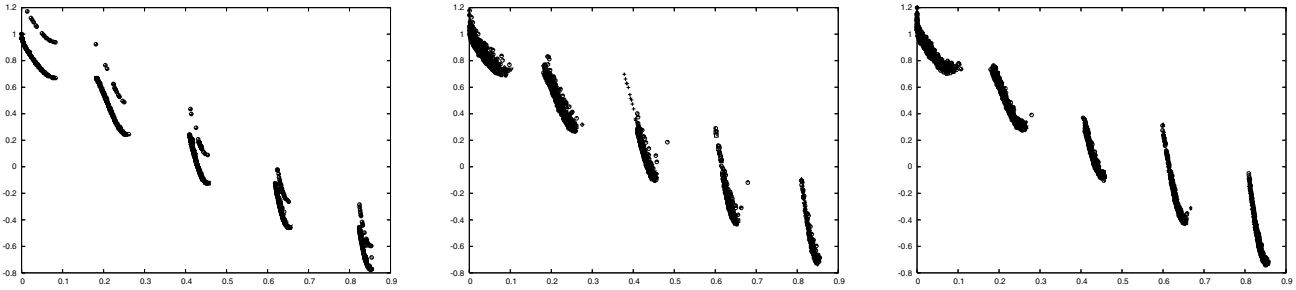

Figure 8. Pareto fronts produced by ST-MOPSO (left), ε-MOEA (center) and NSGA II (right) for the ZDT3's test function.

TSC	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0.934675	0.93291
epsMOEA	0		0.24594
NSGA-II	0	0.0680684	

HV	ST-MOPSO	ε-MOEA	NSGA-II
ST-MOPSO		0	0
epsMOEA	0.019099		0.000320688
NSGA-II	0.026448	0.0076528	

Table 5. Results of the Two Set Coverage and Hyper Volume metrics for the ZDT3's test function.

IGD	ST-MOPSO	ε-MOEA	NSGA-II
Best	**0.000705166**	0.00236056	00163111
Worst	**0.0397353**	0.0225697	0.0231257
Mean	**0.0036884**	0.00842278	0.0065121
Stdev	**0.00707846**	0.00397113	0.00445668
Median	**0.00203803**	0.00804916	0.0059502

SC	ST-MOPSO	ε-MOEA	NSGA-II
Best	**100**	4	1
Worst	0	0	0
Mean	**82.7333**	0.566667	0.0666667
Stdev	25.8082	1.16511	**0.253708**
Median	**88.5**	0	0

Table 6. Results of the Inversed Generational Distance and Sucess Counting metrics for the ZDT3's test function.

is due to the relatively low number of fitness function evaluations considered. When using a larger number of evaluations the two other approaches are able to reach consistently the true Pareto front of the test functions adopted). Note that our ST-MOPSO was the only algorithm able to cover the entire Pareto front of the test problems adopted in our comparative study.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new mechanism to maintain diversity which is based on the use of stripes. The mechanism was incorporated into a multi-objective particle swarm optimizer (MOPSO) in order to validate its effectiveness. The results indicate that the approach is a viable alternative to maintain diversity in a multi-objective evolutionary algorithm (not necessarily a particle swarm optimizer).

As part of our future work, we will extend the approach to handle any number of dimensions (i.e., objectives), since our current version only deals with bi-objective optimization problems. We also intend to test this approach with other types of multi-objective optimization heuristics, such as the artificial immune system [1]. Finally, we are also developing a new metric based on the stripes introduced in this paper. The idea is that this new metric can be used to assess the performance of multi-objective evolutionary algorithms regarding spread and distribution of nondominated solutions.

Acknowledgments

The second author acknowledges support from CONACyT through a scholarship to pursue graduate studies at the Computer Science Section of the Electrical Engineering Department at CINVESTAV-IPN. The last author acknowledges support from CONACyT project number 42435-Y.

7. REFERENCES

[1] Carlos A. Coello Coello and Nareli Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[2] Carlos A. Coello Coello, Gregorio Toscano Pulido, and Maximino Salazar Lechuga. Handling Multiple Objectives With Particle Swarm Optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):256–279, June 2004.

[3] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. Towards a Quick Computation of Well-

Spread Pareto-Optimal Solutions. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 222–236, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.

[4] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[5] Xiaohui Hui, Russell C. Eberhart, and Yuhui Shi. Particle Swarm with Extended Memory for Multiobjective Optimization. In *2003 IEEE Swarm Intelligence Symposium Proceedings*, pages 193–197, Indianapolis, Indiana, USA, April 2003. IEEE Service Center.

[6] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining Convergence and Diversity in Evolutionary Multi-objective Optimization. *Evolutionary Computation*, 10(3):263–282, Fall 2002.

[7] Xiaodong Li. Better Spread and Convergence: Particle Swarm Multiobjective Optimization Using the Maximin Fitness Function. In Kalyanmoy Deb et al., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*, pages 117–128, Seattle, Washington, USA, June 2004. Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.

[8] Mahdi Mahfouf, Min-You Chen, and Derek Arturh Linkens. Adaptive Weighted Particle Swarm Optimisation for Multi-objective Optimal Design of Alloy Steels. In *Parallel Problem Solving from Nature - PPSN VIII*, pages 762–771, Birmingham, UK, September 2004. Springer-Verlag. Lecture Notes in Computer Science Vol. 3242.

[9] K.M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1998.

[10] Sanaz Mostaghim and Jürgen Teich. The role of ε-dominance in multi objective particle swarm optimization methods. In *CEC'2003*, volume 3, pages 1764–1771. IEEE Press, December 2003.

[11] Sanaz Mostaghim and Jürgen Teich. Strategies for Finding Good Local Guides in Multi-objective Particle Swarm Optimization (MOPSO). In *2003 IEEE Swarm Intelligence Symposium Proceedings*, pages 26–33, Indianapolis, Indiana, USA, April 2003. IEEE Service Center.

[12] Sanaz Mostaghim and Jürgen Teich. Covering Pareto-optimal Fronts by Subswarms in Multi-objective Particle Swarm Optimization. In *2004 Congress on Evolutionary Computation (CEC'2004)*, volume 2, pages 1404–1411, Portland, Oregon, USA, June 2004. IEEE Service Center.

[13] K.E. Parsopoulos, D.K. Tasoulis, and M.N. Vrahatis. Multiobjective Optimization Using Parallel Vector Evaluated Particle Swarm Optimization. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications (AIA 2004)*, volume 2, pages 823–828, Innsbruck, Austria, February 2004. ACTA Press.

[14] Dipti Srinivasan and Tian Hou Seow. Particle Swarm Inspired Evolutionary Algorithm (PS-EA) for Multiobjective Optimization Problem. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 4, pages 2292–2297, Canberra, Australia, December 2003. IEEE Press.

[15] Gregorio Toscano Pulido and Carlos A. Coello Coello. Using Clustering Techniques to Improve the Performance of a Particle Swarm Optimizer. In Kalyanmoy Deb et al., editor, *Genetic and Evolutionary Computation–GECCO 2004. Proceedings of the Genetic and Evolutionary Computation Conference. Part I*, pages 225–237, Seattle, Washington, USA, June 2004. Springer-Verlag, Lecture Notes in Computer Science Vol. 3102.

[16] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[17] David A. Van Veldhuizen and Gary B. Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *2000 Congress on Evolutionary Computation*, volume 1, pages 204–211, Piscataway, New Jersey, July 2000. IEEE Service Center.

[18] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[19] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.