A New Mechanism to Maintain Diversity in Multi-Objective Metaheuristics

Mario Alberto Villalobos-Arias[1]
Gregorio Toscano Pulido[2*]
Carlos A. Coello Coello[3†]

[1]Universidad de Costa Rica
Escuela de Matemática, CIMPA
San Pedro de Montes de Oca, Apartado 2060
San José, COSTA RICA
mario.villalobos@ucr.ac.cr

[2]Information Technology Laboratory.
Cinvestav Tamaulipas
Scientific and Technological Park TECNOTAM
Km. 5.5 carretera Cd. Victoria-Soto La Marina
Cd. Victoria, Tamaulipas, 87130, MEXICO
gtoscano@tamps.cinvestav.mx

[3]CINVESTAV-IPN (Evolutionary Computation Group)[‡]
Computer Science Department
Av. IPN No. 2508, Col. San Pedro Zacatenco
México D.F. 07300, MÉXICO
ccoello@cs.cinvestav.mx

October 19, 2010

Abstract

In this paper, a new mechanism to spread the solutions generated by a multi-objective evolutionary algorithm is proposed. This approach is based on the use of stripes that are applied in objective function space

*The second author gratefully acknowledges support from CONACyT project No. 105060.
†The third author gratefully acknowledges support from CONACyT project No. 103570.
‡The third author is also affiliated to the UMI LAFMIA 3175 CNRS at CINVESTAV-IPN

and is independent of the search engine adopted. Additionally, it overcomes some of the drawbacks of other previous proposals such as the ε-dominance method. In order to validate the proposed approach, it is coupled to a multi-objective particle swarm optimizer and its performance is assessed with respect to that of state-of-the-art algorithms, using standard test problems and performance measures taken from the specialized literature. The results indicate that the proposed approach is a viable diversity maintenance mechanism that can be incorporated to any multi-objective metaheuristic used for multi-objective optimization.

Keywords: multi-objective optimization, metaheuristics, Pareto dominance

Mathematics Subject Classification: 90-08, 65K99, 68W20.

1 Introduction

In the real world, there are many problems with two or more (conflicting) objectives which are meant to be optimized simultaneously. A typical example is when one wishes to maximize a certain utility or revenue function, but at the same time, it is desired to minimize, say, an operational cost, which is a conflicting objective. Because of their nature, multi-objective optimization problems (MOPs) have, not one, but a set of solutions (called the *Pareto optimal set*) representing the best possible trade-offs among the objectives. The vectors of the solutions contained in the Pareto optimal set are said to be *nondominated*. The image of the Pareto optimal set (i.e., their corresponding objective function values) is called the *Pareto front*. MOPs have been solved using mathematical programming techniques for a long time [21, 10]. However, because of their ease of use, low need of specific domain information and wide applicability, metaheuristics have become very popular as multi-objective optimization tools in the last few years [3, 6].

Metaheuristics used for solving MOPs normally consist of two main components: (1) a selection mechanism that favors nondominated solutions and (2) a density estimator that allows the algorithm to generate several (different) nondominated solutions in a single run (something normally difficult to achieve with mathematical programming techniques).

For the first component, most multi-objective metaheuristics adopt a selection mechanism based on Pareto optimality [3]. For the second component, a variety of approaches have been proposed, including fitness sharing [7, 12], crowding [9], clustering [26], geographically-based schemes [18], geometrically-based schemes [23], entropy [17], and ϵ-dominance [19], among others. Some of these approaches are very simple to implement (e.g., fitness sharing [7] and clustering [26], which rely on measuring Euclidean distances between pairs of solutions) but are relatively costly (computationally speaking) and rely on problem-specific parameters that are normally difficult to define (e.g., the niche radius) and whose values have a high impact on performance. Others are efficient and

rely on few parameters, but cannot be easily scaled when dealing with more than 2 objectives (e.g., crowding [9], the adaptive grid [18] and entropy [17], which rely on relatively simple concepts derived from geometry or from information theory). Modern density estimators normally require few user-defined parameters, are efficient and can be scaled to more than 2 objectives. From them, it is worth mentioning two particular approaches: the sigma method [23] and ε-dominance [19]. These approaches are still relatively simple, but are also computationally efficient and quite effective. However, they also have limitations, which are related to the shape of the Pareto front of the problem being solved (e.g., ε-dominance always loses the extreme parts of the Pareto front and tends to lose many points in any almost-straight portions of a Pareto front). It is shown that the mechanism proposed here is able to overcome such limitations, while keeping the main advantages of modern density estimators (namely, computational efficiency and the use of few user-defined parameters).

The proposed approach can be coupled to any multi-objective metaheuristic, but for validation purposes, it is coupled here to a multi-objective particle swarm optimizer (MOPSO). The performance of the resulting approach is assessed using standard test problems and performance measures commonly adopted in the specialized literature.

The remainder of this paper is organized as follows. Some basic concepts required to understand the rest of the paper are introduced in Section 2. Section 3 presents the most relevant previous related work. The proposed approach is described in Section 4. Section 5 presents a comparison of the results produced by the proposed approach (coupled to a MOPSO) and two multi-objective evolutionary algorithms (MOEAs) that are representative of the state-of-the-art. Finally, in Section 6, the conclusions of this work and some possible paths for future research are presented.

2 Basic Concepts

Let X be a set and $F : X \longrightarrow I\!R^d$ a given vector function with components $f_i : X \longrightarrow I\!R$ for each $i \in \{1, \ldots, d\}$. The multi-objective optimization problem (MOP) of interest for the purposes of this paper consists of finding: $x^* \in X$ such that

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

where the minimum is understood in the sense of the standard Pareto order in which two vectors in $I\!R^d$ are compared as follows.

If $\vec{u} = (u_1, \ldots, u_d)$ and $\vec{v} = (v_1, \ldots, v_d)$ are vectors in $I\!R^d$, then

$$\vec{u} \preceq \vec{v} \iff u_i \leq v_i \ \forall \ i \in \{1, \ldots, d\}. \tag{2}$$

This relation is a partial order. It is also written as $\vec{u} \prec \vec{v}$ if $\vec{u} \preceq \vec{v}$ and $\vec{u} \neq \vec{v}$. In this case it is said that u *dominates* v. For example, in Figure 1, point B dominates point E.

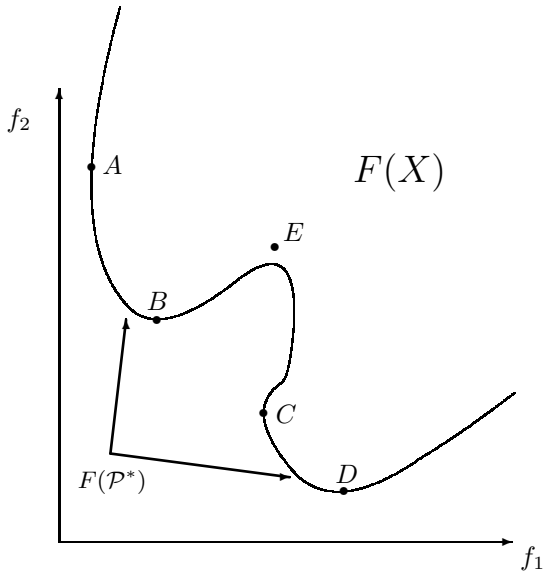


Figure 1: Example of a Pareto front for a bi-objective case.

Definition 2.1 *A point $x^* \in X$ is called a* Pareto optimal solution *for the MOP (2) if there is no $x \in X$ such that $F(x) \prec F(x^*)$. The set*

$$\mathcal{P}^* = \{x \in X \ : \ x \text{ is a Pareto optimal solution}\}$$

is called the Pareto optimal set *for the MOP (2), and its image under F, i.e.*

$$F(\mathcal{P}^*) \ := \ \{F(x) \ : \ x \in \mathcal{P}^*\},$$

is called the Pareto front.

In Figure 1, the Pareto front corresponds to the parts on the boundary of $F(X)$ joining the points A and B, and also the points C and D.

Here, it is said that x *dominates* y when $F(x) \prec F(y)$. Let $Y \subseteq X$ and $y \in Y$. If there is no $x \in Y$, that dominates y, it is said that y is *nondominated* (with respect to Y). Observe that all the elements in the Pareto front are nondominated with respect to X.

Next, the well-known "scalarization" result that will be used later on is presented.

Lemma 2.1 *If $\vec{x}^* \in X$ is a solution of the weighted problem:*

$$\min_{\vec{x} \in X} \sum_{s=1}^{d} w_s f_s(\vec{x}), \ \text{where } w_s > 0 \ \forall s \in \{1, \ldots, d\} \ \text{and} \ \sum_{s=1}^{d} w_s = 1, \qquad (3)$$

then $\vec{x}^* \in \mathcal{P}^*$.

3 Previous Related Work

Particle swarm optimization (PSO) is a relatively recent metaheuristic which is inspired on the flight of a flock of birds or fish seeking food. PSO has been successfully applied to a wide variety of optimization problems [16, 11], but its use in multi-objective optimization started only a few years ago (in the late 1990s). However, and in spite of its recent adoption for solving multi-objective optimization problems, a high number of multi-objective particle swarm optimizers (MOPSOs) exist today (see for example [30, 15, 25, 24, 14, 4]).

Nevertheless, and in spite of this important number of publications on MOPSOs, most of such work has placed little emphasis in aspects such as the importance of the scheme adopted to select leaders [1], and the mechanism adopted to maintain diversity. This last topic is the one that it is dealt with in this paper although, as indicated before, the proposed approach does not use any specific features of PSO (or any other metaheuristic, for that sake) and, therefore, can be coupled to any other multi-objective metaheuristic.

Some researchers have proposed the use of novel mechanisms to maintain diversity in a MOPSO. Two of the most representative approaches in this area are: (1) the sigma method [23] and (2) ε-dominance (a relaxed form of Pareto dominance that can be used to regulate convergence of a MOEA, when adopted as an archiving technique [19]) [22]. Next, these two approaches are briefly discussed.

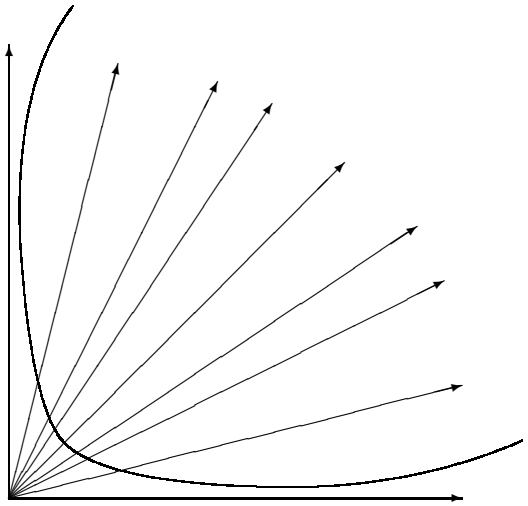


Figure 2: This figure illustrates a situation that causes problems to the sigma method proposed by Mostaghim et al. [23].

The sigma method uses the vector represented by the evaluation of $F(\vec{x})$ of the particle $\vec{x}$, and the leader of this particle is the individual in the elite set[1] 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 [23]). 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.

Let us consider again the MOP (2). The concept of ε-dominance [19] refers to a relaxed form of Pareto dominance. A decision vector x_1 is said to ε-dominate a decision vector x_2 for some $\varepsilon > 0$ iff: $f_i(x_1)/(1+\varepsilon) \leq f_i(x_2), \forall i = 1, \ldots, d$, and $f_i(x_1)/(1+\varepsilon) < f_i(x_2)$; for at least one $i = 1, \ldots, d$ (d is the total number of objective functions of the problem). It is worth noting that ε is a user-defined parameter.

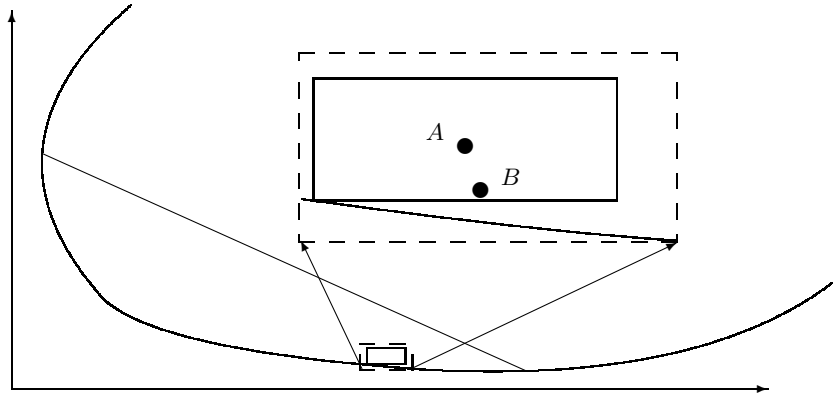


Figure 3: An example in which the ε-dominance approach retains the wrong point.

[1]Most modern MOEAs use an *elite set* in which the solutions that are "globally" nondominated (i.e., nondominated with respect to all the individuals that have been processed so far) are stored. This set is theoretically required in order to guarantee convergence [29].

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 (i.e., the *elite set*). 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, the proposed approach will retain point B.

4 Stripes-based mechanism

Throughout the remainder of this paper the functions $f_1, \ldots, f_d$ are assumed to be bounded below.

First, a result and a definition that are part of the core idea of the proposed approach are presented.

Lemma 4.1 *Let $\vec{x}^1, \vec{x}^2, \ldots, \vec{x}^d \in X$ be the minimizers of the functions $f_1, f_2, \ldots, f_d$ respectively. 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 4.1 is trivial and is, therefore, omitted here. The lemma is illustrated in Figure 4, 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,

Definition 4.1 *Let $X_1, \ldots, X_d \in I\!R^d$ then the* **convex hull** *of these vectors is*

$$CH(X_1, \ldots, X_d) = \left\{ \sum_{i=1}^{d} \alpha_i X_i / \sum_{i=1}^{d} \alpha_i = 1, \alpha_i \geq 0, \alpha_i \in I\!R \right\}. \tag{4}$$

The core idea of the approach proposed in this paper (called "stripes") is that the convex hull generated by the points $F(\vec{x}^1), F(\vec{x}^2), \ldots, F(\vec{x}^d)$ (defined in Lemma 4.1) is "similar" to the Pareto front (see Figures 5 and 6). Thus, several points can be used (which will be called stripe centers) uniformly distributed along this convex hull, and the individuals of the population are assigned to the nearest stripe center. This way, the individuals are being distributed in several stripes determined by the stripe centers (see Figure 5). Now, if an upper bound is set 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, the notion of clustering is adopted, but the center of each cluster is fixed and uniformly distributed along the convex hull, as shown in Figures 5 and 6 (the small circles are the centers of the clusters).

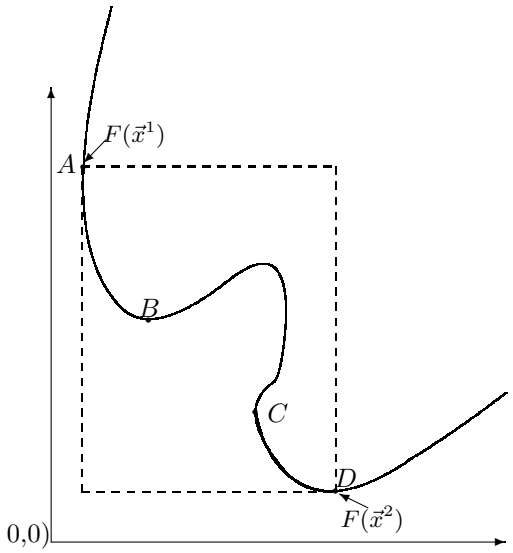


Figure 4: $F(\mathcal{P}^*)$ is contained in the "hyper-box" defined by $F(\vec{x}^1),\ F(\vec{x}^2)$

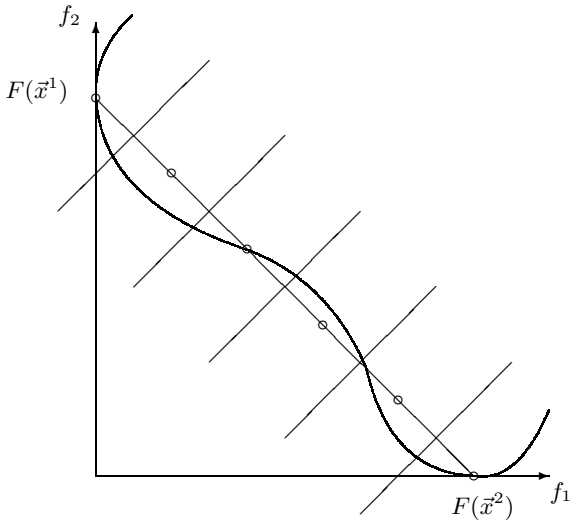


Figure 5: Graphical representation of the stripes proposed in this paper.

The stripe center set can be computed using:

$$\left\{ X_{i_1,\ldots,i_d} = \sum_{j=1}^{d} \frac{i_j F(\vec{x}^i)}{nl} \;/\; i_1 + \cdots + i_d = nl,\; i_1, \ldots, i_d \in I\!N \cup \{0\} \right\} \qquad (5)$$

where $nl \geq 1$, is a parameter provided by the user which refers to the number of points in each edge of the convex hull.

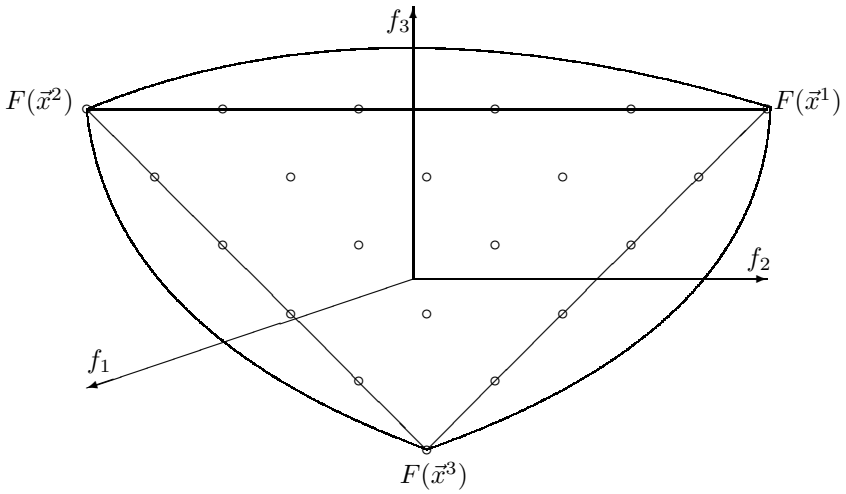


Figure 6: Distribution of the stripes center for $d = 3$, $nl = 6$

Special case $d = 2$

In the case $d = 2$, some simplifications can be done, as shown next.

In this case the stripe center set can be computed using

$$\left\{ X_i = \frac{iF(\vec{x}^1) + (ns - 1 - i)F(\vec{x}^2)}{ns - 1}, i \in \{0, 1, \ldots, ns - 1\} \right\}, \qquad (6)$$

where ns is the number of stripes, which is equal to nl.

In the case in which there are only two objective functions, $d = 2$, a rotation can be applied 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 all the elements need to be rotated. Further, if $F^r(\vec{x}) = (f_1^r(\vec{x}), f_2^r(\vec{x}))$ are the rotated coordinates of

$F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}))$, then

$$\begin{aligned} f_1^r(\vec{x}) &= \cos(\theta) f_1(\vec{x}) - \sin(\theta) f_2(\vec{x}) \\ f_2^r(\vec{x}) &= \sin(\theta) f_1(\vec{x}) + \cos(\theta) f_2(\vec{x}) \end{aligned} \tag{7}$$

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

$$h = \frac{f_1^r(\vec{x}^2) - f_1^r(\vec{x}^1)}{ns - 1}, \quad \text{and} \quad h_{\vec{x}} = \frac{f_1^r(\vec{x}) - f_1^r(\vec{x}^1)}{h}. \tag{8}$$

Then

$$\text{stripe}(\vec{x}) = \begin{cases} 1 & \text{if} \quad h_{\vec{x}} < 0.5 \\ [\![\, h_{\vec{x}} + 1.5]\!] & \text{if} \quad 0.5 \leq h_{\vec{x}} < ns - 0.5 \\ ns & \text{if} \quad h_{\vec{x}} \geq ns - 0.5, \end{cases} \tag{9}$$

where $[\![r]\!]$ denotes the integer part of $r \in I\!R$.

This procedure to assign the stripe is simpler than the method for calculating distances to the stripe center and for computing the minimum of these distances.

To illustrate the way in which the proposed approach works, Figure 7 shows an example for a problem with two objectives. In the figure, it can be seen that the approach (which was coupled to the MOPSO proposed in [4]) generates a good representation of the Pareto front.

4.1 A MOPSO with Stripes

In order to validate the effectiveness of the proposed approach to maintain diversity, the MOPSO proposed in [4] is used as the search engine. However, in this case, the diversity maintenance scheme are the stripes proposed in this paper instead of the adaptive grid originally adopted in [4]. This MOPSO with stripes is called ST–MOPSO.

The proposal consists of using one leader in each stripe and to compute a weighted sum determined by the points $F(\vec{x}^1), F(\vec{x}^2), \ldots, F(\vec{x}^d)$ defined in Lemma 4.1, to select the leaders, where the leader of a stripe is the point that minimizes this weighted sum.

To compute the parameter of the scalarization, the coefficients of the normal vector $\vec{n}$ of the "affine subspace" (hyper–plane) that contains the points $F(\vec{x}^1), F(\vec{x}^2), \ldots, F(\vec{x}^d)$, are used.

Thus, for $d = 2$ the normal vector is:

$$\vec{n} = (|f_2(\vec{x}^1) - f_2(\vec{x}^2)|, |f_1(\vec{x}^1) - f_1(\vec{x}^2)|) \tag{10}$$

For the case $d = 3$ the vector product will be used as follows.

Let $(a, b, c) = [F(\vec{x}^1) - F(\vec{x}^2)] \times [F(\vec{x}^1) - F(\vec{x}^3)]$ then

$$\vec{n} = (|a|, |b|, |c|) \tag{11}$$

Algorithm 1 ST-MOPSO Algorithm

$\vec{gbest} \leftarrow \vec{x_0}$
for $i = 0$ to $nparticles$ **do**
  $\vec{pbest_i} \leftarrow \vec{x_i} \leftarrow initialize\_randomly()$
  $fitness_i \leftarrow f(\vec{x_i})$
  if $\neg\exists\ \vec{y^*}\ \in\ GBEST \mid \vec{f}(\vec{y}*) \preceq \vec{f}(\vec{fitness_i}^*)$ **then**
    $GBEST \leftarrow GBESTU\vec{x_i}$
  end if
end for
repeat
  for $i = 0$ to $nparticles$ **do**
    $gbest \leftarrow GBEST_{U(0,|GBEST|)}$
    for $d = 0$ to $ndimensions$ **do**
      $velocity_{id} \leftarrow W \times velocity_{id} + C_1 \times U(0,1) \times (pbest_{id} - x_{id}) + C_2 \times U(0,1) \times (gbest_{id} - x_{id})$
      $x_{id} \leftarrow x_{id} + velocity_{id}$
    end for
    if flip(1/10) **then**
      perturb particle i
    end if
    if |GBEST| < 5) **then**
      perturb 10% of the range of one dimension (randomly selected) of particle i
    end if
    $fitness_i \leftarrow f(\vec{x_i})$
    if $fitness_i$ is nondominated with respect to $f(\vec{pbest_i})$ **then**
      $\vec{pbest_i} \leftarrow \vec{x_i} \in$
    end if
    if $\neg\exists\ \vec{y^*}\ \in\ GBEST \mid \vec{f}(\vec{y}*) \preceq \vec{f}(\vec{fitness_i}^*)$ **then**
      $GBEST \leftarrow GBESTU\vec{x_i}$
    end if
  end for$GBEST \leftarrow stripes - approach(GBEST)$
until Termination criterion

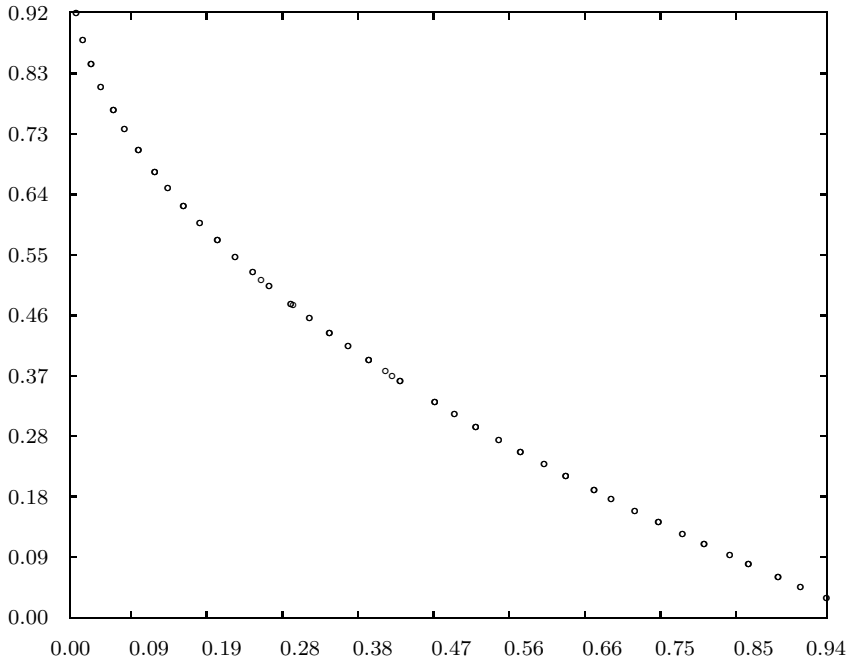


Figure 7: An example of the type of distribution of nondominated solutions produced by the proposed approach.

In the other cases the Gram-Schmidt orthogonalization process will be used to obtain the normal vector or the orthogonal projection of some special vector on the "Affine Subspace".

Thus, the leader of a stripe is the element that minimizes

$$\vec{n} \cdot F = \sum_{i=1}^{d} n_i f_i(\vec{x}), \tag{12}$$

with $\vec{n} = (n_1, \ldots, n_d)$.

From Lemma 2.1, if the leader is a minimal point of (12) it is in $\mathcal{P}^*$. Thus, this scheme for selecting the leaders makes sense.

However, in the case in which $d = 2$, the rotated coordinates $F^r(\vec{x})$ are available and the parameters of the scalarization can be taken as:

$$n_1 = \sin(\theta), \text{ and } n_2 = \cos(\theta), \tag{13}$$

the coefficients of the rotation. In this case, the leader of a stripe is the particle in the elite set that minimizes f_2^r (the y-coordinate of F^r).

Algorithms	**P**	G_{max} **needed to reach the required FFEs**					**P**
		400	800	1,200	1,600	2,000	
ST-MOPSO	40	9	19	29	39	49	100
PAPSO	40	9	19	29	39	49	100
MOPSO	40	9	19	29	39	49	100
NSGA-II	100	3	7	11	15	19	100
ε-MOEA	100	150	350	550	750	950	$\approx$ 100

Table 1: Generic parameters used by the algorithms compared. FFEs = fitness function evalations. G_{max} = maximum number of generations (iterations)

5 Comparison of Results

Several test functions were taken from the specialized literature to validate the proposed approach. First, the results obtained by ST–MOPSO for three bi-objective test problems will be included. The obtained 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 [9] and the ε-MOEA [8] and two PSO-based approaches, the MOPSO from [4] and a slight modification of that MOPSO that is called PAPSO. PAPSO uses Pareto dominance in order to compare solutions. It also uses an archive which serves as an explicit form of elitism in order to store the nondominated solutions found so far in a single execution. However, it does not use any explicit approach to maintain diversity in the archive.

After that, an example which considers a MOP with 3 objective functions (see subsection 5.4) is presented. The results will be, again, compared with respect to those obtained by PAPSO, MOPSO, NSGA-II and ε-MOEA.

Each approach performed $2,000$ fitness function evaluations for the test functions, in the results shown next. However, the solutions at every 400 fitness function evaluations are pulled out in order to perform a more detailed analysis of the proposal (i.e., the results of the algorithms when they are using 400, 800, $1,200$, $1,600$ and $2,000$ fitness evaluations of every test problem included in this study are analyzed). The results shown correspond to 30 independent runs.

Table 1 summarizes the general parameter settings adopted for all the algorithms compared. In Table 1, P refers to the population at each generation, G_{max} is the total number of generations (or iterations) to be performed. Note that all the algorithms perform the same number of objective function evaluations for all test problems (the fitness functions computed for the initialization process are also counted). NP is the number of solutions expected by each algorithm; in the case of ε-MOEA, this parameter is controlled by the value of $\vec{\varepsilon}$ which is shown in Table 2.

Table 2 summarizes the specific parameter settings adopted for all the algorithms compared.

In order to allow a quantitative comparison of results the following *perfor-*

Algorithms	Variation Parameters
ST-MOPSO	$C_1 = 1.4962$,$C_2 = 1.4962$, $W = 0.7298$, $ns = 100$, perturbation ratio $1/10$
PAPSO	$C_1 = 1.4962$,$C_2 = 1.4962$, $W = 0.7298$, perturbation ratio $1/nvars$
MOPSO	$C_1 = 1.4962$,$C_2 = 1.4962$, $W = 0.7298$, perturbation ratio $1/nvars$
NSGA-II	Simulated binary crossover ($P_c = 0.8$, $\eta_c = 15$), parameter-based mutation ($P_m = 1/nvars$,$\eta_c = 20$)
ε-MOEA	Simulated binary crossover ($P_c = 0.8$, $\eta_c = 15$), parameter-based mutation ($P_m = 1/nvars$,$\eta_c = 20$), $\varepsilon_{ZDT1} = \varepsilon_{ZDT2} = \varepsilon_{ZDT3} = [0.007, 0.007]^T$, $\varepsilon_{viennet} = [0.11, 0.11, 0.11]^T$

Table 2: Specific parameters used by the algorithms compared.

mance measures were adopted: standard deviation of the crowding distances [13], spread [9], inverted generational distance [27], ratio of the hypervolume [20], success counting (which is a variation of the performance measure called "error ratio" [27]) and the number of nondominated points found by the final population.

The definition of each of these performance measures is presented next.

5.1 Standard Deviation of the Crowding Distances (SDC)

This performance measure computes the standard deviation of the crowding distance in order to give more information about the distribution of the solutions:

Definition 5.1 *The SDC performance measure is defined as:*

$$SDC = \sqrt{\frac{1}{|F|}\sum_{i=1}^{|F|}(d_i - \overline{d_i})^2} \tag{14}$$

$0 \leq SDC \leq \infty$. The lower the value of SDC, the better the distribution of vectors in F. d_i is the mean value of all d_i. Again, a perfect distribution, that is $SDC = 0$, means that d_i is constant for all i.

5.2 Spread

Deb et al. [9] proposed the performance measure Δ with the idea of measuring both progress towards the Pareto-optimal front and the extent of spread. To this end, if P is a subset of the Pareto-optimal front, Δ is defined as follows

Definition 5.2 *The Δ performance measure is defined as:*

$$\Delta = \frac{\sum_{i=1}^{m} d_i^e + \sum_{i=1}^{|F|} |d_i - \overline{d}|}{\sum_{i=1}^{m} d_i^e + |F|\overline{d}}. \tag{15}$$

where d_i^e denotes the distance between the i-th coordinate for both extreme points in P and F, and d_i measures the distance of each point in F to its closest point in F . For the experiments, the crowding distance for d_i is used (see [6] for more details on this distance). Nevertheless, other types of measures could be used for d_i. From the above definition, it is easy to conclude that $0 \leq \Delta \leq 1$ and the lower the Δ value, the better the distribution of solutions. A perfect distribution, that is $\Delta = 0$, means that the extreme points of the Pareto-optimal front have been found and d_i is constant for all i.

Inverted Generational Distance (IGD)

The concept of generational distance (GD) was introduced by Van Veldhuizen & Lamont [27, 28] as a way of estimating how far are the elements in the Pareto front produced by the algorithm from those in the true Pareto front of the problem.

Definition 5.3 *Generational distance is defined as:*

$$GD = \frac{\left(\sum_{i=1}^{N} d_i^2\right)^{1/2}}{N} \tag{16}$$

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 vectors 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" are these solutions from the global Pareto front of the problem. Here, an "inverted" generational distance measure (IGD) was implemented, using as a reference the true Pareto front, and each of its elements are compared with respect to the front produced by an algorithm. In this way, IGD computes how far are the elements of the true Pareto front, from those in the Pareto front produced by the algorithm being analyzed. Computing this "inverted" generational distance value reduces the bias that can arise when an algorithm does not fully cover the true Pareto front. It is worth noting that, since this performance measure requires having the true Pareto front of the problem, it cannot be used in problems in which the true Pareto front is not known and is very costly (computationally speaking) to compute it (e.g., in large instances of NP-Complete problems).

Ratio of the Hypervolume (HVR)

The ratio of the hypervolume (HVR) performance measure was introduced in [20], and is based on the "size of the space covered" performance measure, which was originally proposed in Zitzler's PhD thesis [31].

Let $\mathcal{P}_A^*$ be the solution (nondominated elements) of an algorithm A . First, the hypervolume needs to be computed[2] for the region determined by the image $(F(\mathcal{P}_A^*))$ of the solution of the algorithm A and the origin.

For example, a vector $\vec{x} \in \mathcal{P}_A^*$ for a two-objective problem defines a rectangle bounded by a reference point W and $(f_1(\vec{x}), f_2(\vec{x}))$. The area of the union of all such rectangles defined by each vector in $\mathcal{P}_a^*$ is the hypervolume. Then the hypervolume $V(\mathcal{P}_A^*)$ determined by the algorithm A is defined as:

$$V(\mathcal{P}_A^*) := \text{Hypervolume}\left\{\bigcup_{\vec{x}\in\mathcal{P}_A^*} a_{\vec{x}}\right\}, \tag{17}$$

where $a_{\vec{x}}$ is the hyperbox determined by the components of $F(\vec{x})$ and the W reference point. V depends strongly of the choice of W. A set of nondominated solutions can produce different values of V for different choices of W. Therefore, the use of this metric makes difficult to compare different values for V in different iterations. The use of the ratio of the hypervolume is a way to overcome this problem.

Definition 5.4 *Let Pop_A and $\mathcal{P}_{PF}^*$ be the solution of an algorithm A and the Pareto front PF of the function at hand, respectively; Then the performance measure HVR is defined as*

$$HVR(A, PF) = \frac{V(\mathcal{P}_A^*)}{V(\mathcal{P}_{PF}^*)} \tag{18}$$

In this performance measure, the maximum value of HVR should be 1 when A is equal to PF and $HVR(A, PF) < 1$, otherwise.

Success Counting (SC)

The *success counting* measure is defined based on the idea of the measure called *Error Ratio* proposed by Van Veldhuizen [27] 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, the number of vectors (in the current set of nondominated vectors available) that are members of the Pareto optimal set are counted:

Definition 5.5 *The* success counting *measure SC is defined by*

$$SC = \sum_{i=1}^{N} s_i, \tag{19}$$

[2]The hypervolume is the area under the curve for the 2-dimensional case or the volume under the surface for the 3-dimensional case.

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 the 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 nondominated solutions that each algorithm produced. It is worth noting that, as in the case of IGF, this performance measure requires having the true Pareto front of the problem.

Number of points

It shows us how far the number of solutions found is from the maximum capacity of the repository. In all the experiments, the repository was defined with a capacity of 100 points. So, the closer to 100 that an algorithm gets, the better the value of this performance measure.

5.3 Performance in Problems with 2 Objective Functions

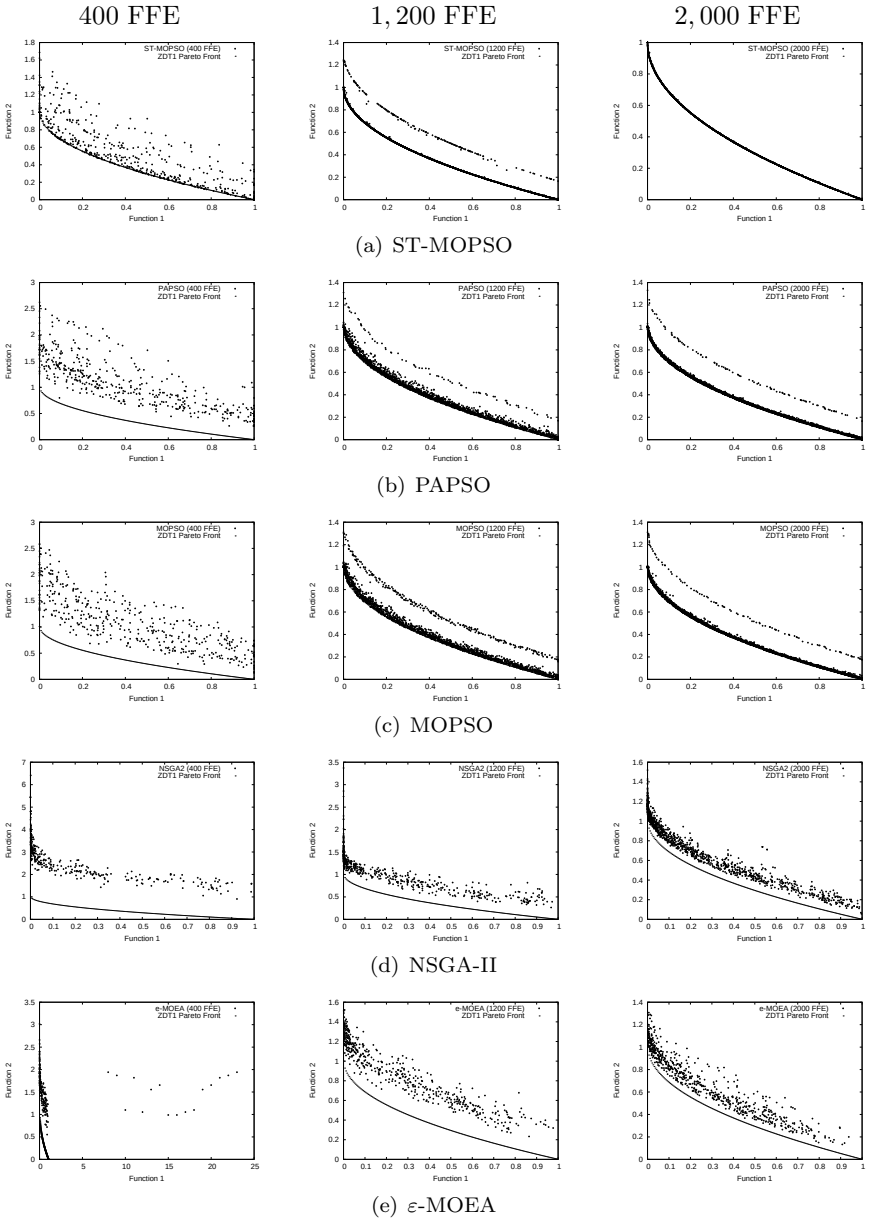
Now the results obtained for the following three test functions taken from [32] are shown:

5.3.1 ZDT1's test function

$$\begin{aligned} \text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\ f_1(\vec{x}) &= x_1 \\ f_2(\vec{x}) &= g(\vec{x})\ h(f_1(\vec{x}), g(\vec{x})) \\ g(\vec{x}) &= 1 + 9 \sum_{i=2}^{m} \frac{x_i}{(m-1)}, \\ h(x, y) &= 1 - \sqrt{\frac{x}{y}} \end{aligned} \tag{20}$$

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

Figure 8 shows the graphical results produced by ST-MOPSO, PAPSO, 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–hand side image of Figure 8).



(a) ST-MOPSO

(b) PAPSO

(c) MOPSO

(d) NSGA-II

(e) ε-MOEA

Figure 8: Solutions produced by 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the ZDT1 test function (using 400, 1200, and 2000 test function evaluations).

Box-plot graphics shown in Figures 9 and 10 depict the behavior of 1) ST-MOPSO, 2) PAPSO, 3) MOPSO, 4) NSGA-II and 5) ε-MOEA for the ZDT1

test function regarding the six performance measures adopted for this study. In order to study the spread behavior of such algorithms, the results of the SDC and Spread performance measures need to be analyzed (see Figures 9(a) and 9(b)). From these results, it should be clear that the proposed approach could distribute its solutions even in early stages of the search. This would not be useful if the proposed approach were not converging to the true Pareto front. However, when analyzing the results obtained by the performance measures IGD, HVR, SC and number of points (see Figures 9(c), 10(a), 10(b) and 10(c)), it is found that ST-MOPSO could find both more nondominated solutions with respect to the true Pareto front and could also distribute better such solutions along the Pareto front as it is reflected by the HVR and IGD performance measures. Therefore, it can be said that ST-MOPSO outperformed the other algorithms with respect to which it was compared, regarding all the performance measures adopted.

By looking at the Pareto fronts produced by each algorithm in this test function, it should be clear that ST-MOPSO 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 8). Also, it can be seen that PAPSO and MOPSO could reach the true Pareto front. However, this is not completely true, since if the SC performance measure shown in Figure 10(b) is analyzed, it can noted that none of these two algorithms could find more than 40 nondominated solutions with respect to the true Pareto front.

5.3.2 ZDT2's test function

$$\begin{aligned} \text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\ f_1(\vec{x}) &= x_1 \\ f_2(\vec{x}) &= g(\vec{x})\ h(f_1(\vec{x}), g(\vec{x})) \\ g(\vec{x}) &= 1 + 9\sum_{i=2}^{m} \frac{x_i}{(m-1)} \\ h(x,y) &= 1 - \left(\frac{x}{y}\right)^2 \end{aligned} \tag{21}$$

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

Figure 11 shows the graphical results produced by ST-MOPSO, PAPSO, MOPSO, the NSGA-II [9] and ε-MOEA [5] in the second test function adopted (the outputs of the 30 independent runs of each algorithm were combined in a single file in order to generate the plots). From this figure, it can be seen that the PSO-based algorithms generated solutions closer to the true Pareto front than NSGA-II and ε-MOEA. When the PSO-based approaches are compared, it should be clear than the ST-MOPSO had a better performance. This graphical-based conclusion can be confirmed if the comparison of results shown in Figures 12 and 13 is analyzed. When the spread behavior shown in Figures 12(a) and 12(b) (SDC and Spread performance measures, respectively) is

observed, it can be noted that the PSO-based approaches presented a better distribution of solutions. However, the proposed approach was the algorithm which showed the best performance for the Spread performance measure. If the IGD and HVR performance measures are analyzed (see Figures 12(c) and 13(a), respectively) it can be easily noted that ST-MOPSO, PAPSO and MOPSO outperformed both NSGA-II and ε-MOEA for this test function. Also, it should be noted that the proposed approach converged to the true Pareto front after performing only $1,200$ fitness function evaluations while the other PSO-based approaches converged after performing $2,000$ fitness function evaluations. The SC performance measure shown in Figure 13(b) confirms the superiority of the proposed approach. Therefore, it can be stated that, as in the previous example, the performance of ST-MOPSO outperformed the other approaches compared for this test function.

5.3.3 ZDT3's test function

$$\begin{aligned} \text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x})) \\ f_1(\vec{x}) &= x_1 \\ f_2(\vec{x}) &= g(\vec{x})\ h(f_1, g) \\ g(\vec{x}) &= 1 + 9 \sum_{i=2}^{m} \frac{x_i}{(m-1)} \\ h(x, y) &= 1 - \sqrt{\frac{x}{y}} - \frac{x}{y} \sin(10\pi x) \end{aligned} \tag{22}$$

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

Figure 14 shows the graphical results produced by ST-MOPSO, PAPSO, MOPSO, the NSGA-II, and ε-MOEA in the third test function chosen (the outputs of the 30 independent runs of each algorithm were combined in a single file in order to generate the plots). From such figure, it can be seen that the proposed ST-MOPSO generated some points outside the true Pareto front in some of the runs. However, when looking at the graphical output generated with $1,200$ fitness function evaluations, it is clear that the proposed ST-MOPSO converged in most of its runs to the true Pareto front. The other PSO-based algorithms converged reasonably well and NSGA-II and ε-MOEA produced a considerably large number of solutions outside the true Pareto front of the problem.

Figures 15 and 16 show the comparison of results among the three algorithms considering the performance measures previously described. In this test function, the Spread performance measure shown in Figure 15(b) does not lead to any conclusive result. This seems to be mainly due to the disconnected form of the Pareto Front. However, when analyzing the rest of the performance measures, it can observed that the approach proposed here could reach the Pareto front since it produced good results for the IGD and the HVR performance

measures (see Figures 15(c) and 16(a)); it also produced a better number nondominated solutions than the other algorithms (see Figures 16(b) and 16(c)). Therefore, it can be said that the proposed approach could reach the true Pareto front while the others did not. Also, as well as in the previous example, it should be clear that the proposed approach cold reach the true Pareto front after performing only $1,200$ fitness function evaluations.

From these performance measures, it can be stated that the proposed approach outperformed the other algorithms.

Summarizing the results obtained for the bi-objective problems, it can be seen that the performance of the proposed ST-MOPSO was the best with respect to all the performance measures adopted. 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, mainly because of 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.

5.4 An example with 3 objective functions

Here, an example with 3 objectives is introduced, the so–called Viennet test function [3]:

$$\begin{aligned} \text{Minimize} \quad & (f_1(\vec{x}), f_2(\vec{x}), f_3(\vec{x})) \\ f_1(\vec{x}) &= x_1^2 + (x_2 - 1)^2 \\ f_2(\vec{x}) &= x_1^2 + (x_2 + 1)^2 + 1 \\ f_3(\vec{x}) &= (x_1 - 1)^2 + x_2^2 + 2 \end{aligned} \tag{23}$$

Figure 17 shows the graphical representation of the solution produced by ST-MOPSO (only the nondominated elements are shown), PAPSO, MOPSO, NSGA-II and ε-MOEA for this example (the outputs of the 30 independent runs of each algorithm were combined in a single file in order to generate the plots).

Figures 18 and 19 show the comparison of results among the three algorithms considering the performance measures previously described. Apparently, the poor values obtained by ε-MOEA are due to a well-known limitation of the ε-dominance mechanism (it loses the extreme parts of the Pareto front). Therefore, it is unable to properly cover the entire Pareto front. It can also be observed that for the SDC and Spread performance measures (see Figures 18(a) and 18(b), respectively), the NSGA-II presented better values than the other algorithms. However, it is important to note that NSGA-II did not approach the true Pareto front as well as ST-MOPSO as can be easily seen in the values of IGD, HVR and SC (see Figures 18(c), 19(a) and 19(b), respectively). When

comparing the PSO-based approaches, it can be noted that their performances are very similar. However, the proposed approach obtained better results for both the IGD and SC performance measures (see Figures 18(c) and 19(b)).

Summarizing, if the six performance measures and the graphical representation are considered, it can be concluded that ST-MOPSO outperformed the other algorithms for this test problem.

6 Conclusions and Future Work

In this paper, a new mechanism to maintain diversity has been proposed, based on the use of stripes. As an example this 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 metaheuristic (not necessarily a particle swarm optimizer).

Some of the possible future work is to extend the approach so that it can handle any number of objectives, since the current version only deals with optimization problems having 2 or 3 objectives. It is also desirable to test this approach with other types of multi-objective optimization heuristics, such as the artificial immune system [2]. Finally, a new performance measure based on the stripes introduced in this paper could also be developed. The idea is that this new performance measure could be used to assess the performance of multi-objective metaheuristics regarding the spread and distribution of the nondominated solutions that they generate.

References

[1] Jürgen Branke and Sanaz Mostaghim. About Selecting the Personal Best in Multi-Objective Particle Swarm Optimization. In Thomas Philip Runarsson, Hans-Georg Beyer, Edmund Burke, Juan J. Merelo-Guervós, L. Darrell Whitley, and Xin Yao, editors, *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference*, pages 523–532. Springer. Lecture Notes in Computer Science Vol. 4193, Reykjavik, Iceland, September 2006.

[2] 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.

[3] Carlos A. Coello Coello, Gary B. Lamont, and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, September 2007. ISBN 978-0-387-33254-3.

[4] 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.

[5] K. Deb, M. Mohan, and S. Mishra. Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions. In C.M. Fonseca, P.J. Fleming, E. Zitzler, K. Deb, and L. 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.

[6] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[7] Kalyanmoy Deb and David E. Goldberg. An Investigation of Niche and Species Formation in Genetic Function Optimization. In J. David Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, San Mateo, California, June 1989. George Mason University, Morgan Kaufmann Publishers.

[8] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. Evaluating the ϵ-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evolutionary Computation*, 13(4):501–525, Winter 2005.

[9] 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.

[10] Matthias Ehrgott. *Multicriteria Optimization*. Springer, Berlin, second edition, 2005. ISBN 3-540-21398-8.

[11] Andries P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, Ltd, 2005. ISBN 978-0-470-09191-3.

[12] David E. Goldberg and Jon Richardson. Genetic algorithm with sharing for multimodal function optimization. In John J. Grefenstette, editor, *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pages 41–49, Hillsdale, New Jersey, 1987. Lawrence Erlbaum.

[13] Alfredo G. Hernández-Díaz, Luis V. Santana-Quintero, Carlos A. Coello Coello, and Julián Molina. Pareto-adaptive ϵ-dominance. *Evolutionary Computation*, 15(4):493–517, Winter 2007.

[14] V.L. Huang, P.N. Suganthan, and J.J. Liang. Comprehensive learning particle swarm optimizer for solving multiobjective optimization problems. *International Journal of Intelligent Systems*, 21(2):209–226, February 2006.

[15] Kazuhiro Izui, Shinji Nishiwaki, Masataka Yoshimura, Masahiko Nakamura, and John E. Renaud. Enhanced multiobjective particle swarm optimization in combination with adaptive weighted gradient-based searching. *Engineering Optimization*, 40(9):789–804, September 2008.

[16] James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco, California, 2001.

[17] Hajime Kita, Yasuyuki Yabumoto, Naoki Mori, and Yoshikazu Nishikawa. Multi-Objective Optimization by Means of the Thermodynamical Genetic Algorithm. In Hans-Michael Voigt, Werner Ebeling, Ingo Rechenberg, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature—PPSN IV*, pages 504–512. Springer-Verlag. Lecture Notes in Computer Science No. 1141, Berlin, Germany, September 1996.

[18] Joshua D. Knowles and David W. Corne. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2):149–172, 2000.

[19] 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.

[20] Xiaodong Li, Jürgen Branke, and Michael Kirley. On Performance Metrics and Particle Swarm Methods for Dynamic Multiobjective Optimization Problems. In *2007 IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 576–583, Singapore, September 2007. IEEE Press.

[21] Kaisa M. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1999.

[22] Sanaz Mostaghim and Jürgen Teich. The Role of ε-dominance in Multi Objective Particle Swarm Optimization Methods. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 3, pages 1764–1771, Canberra, Australia, December 2003. IEEE Press.

[23] 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.

[24] Margarita Reyes-Sierra and Carlos A. Coello Coello. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3):287–308, 2006.

[25] C. H. Tan, C. K. Goh, K. C. Tan, and A. Tay. A Cooperative Coevolutionary Algorithm for Multiobjective Particle Swarm Optimization. In *2007 IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 3180–3186, Singapore, September 2007. IEEE Press.

[26] 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.

[27] 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.

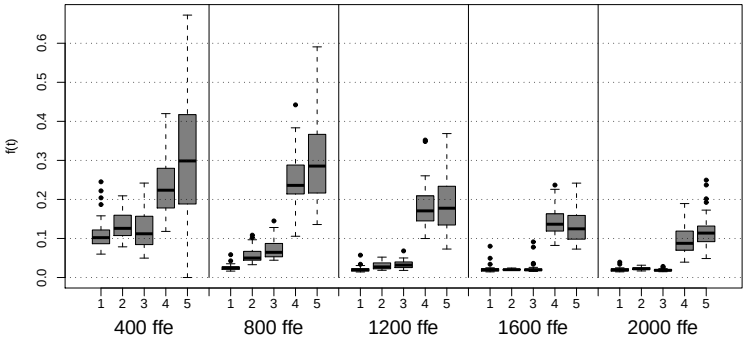
[28] David A. Van Veldhuizen and Gary B. Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *2000 Congress on Evolutionary Computation*, volume 1, pages 204–211, Piscataway, New Jersey, July 2000. IEEE Service Center.

[29] Mario Villalobos-Arias, Carlos A. Coello Coello, and Onésimo Hernández-Lerma. Asymptotic Convergence of Metaheuristics for Multiobjective Optimization Problems. *Soft Computing*, 10(11):1001–1005, September 2006.

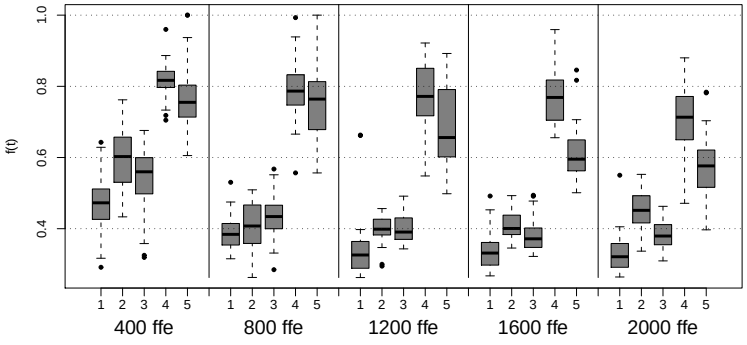
[30] Junjie Yang, Jianzhong Zhou, Li Liu, and Yinghai Li. A novel strategy of pareto-optimal solution searching in multi-objective particle swarm optimization (MOPSO). *Computers & Mathematics with Applications*, 57(11–12):1995–2000, June 2009.

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

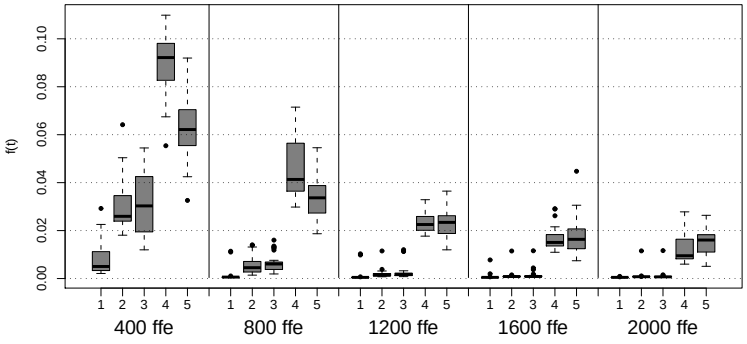
[32] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.



(a) SDC

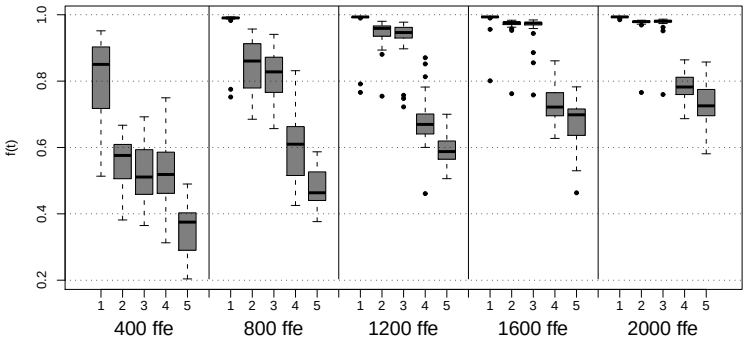


(b) Spread

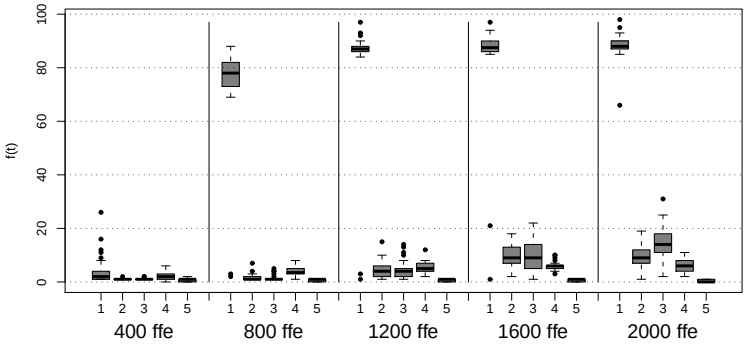


(c) IGD

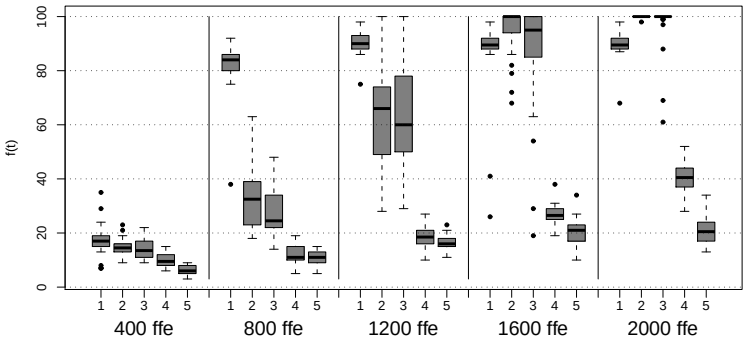
Figure 9: Box-plots produced from the results of 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the ZDT1 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) HVR

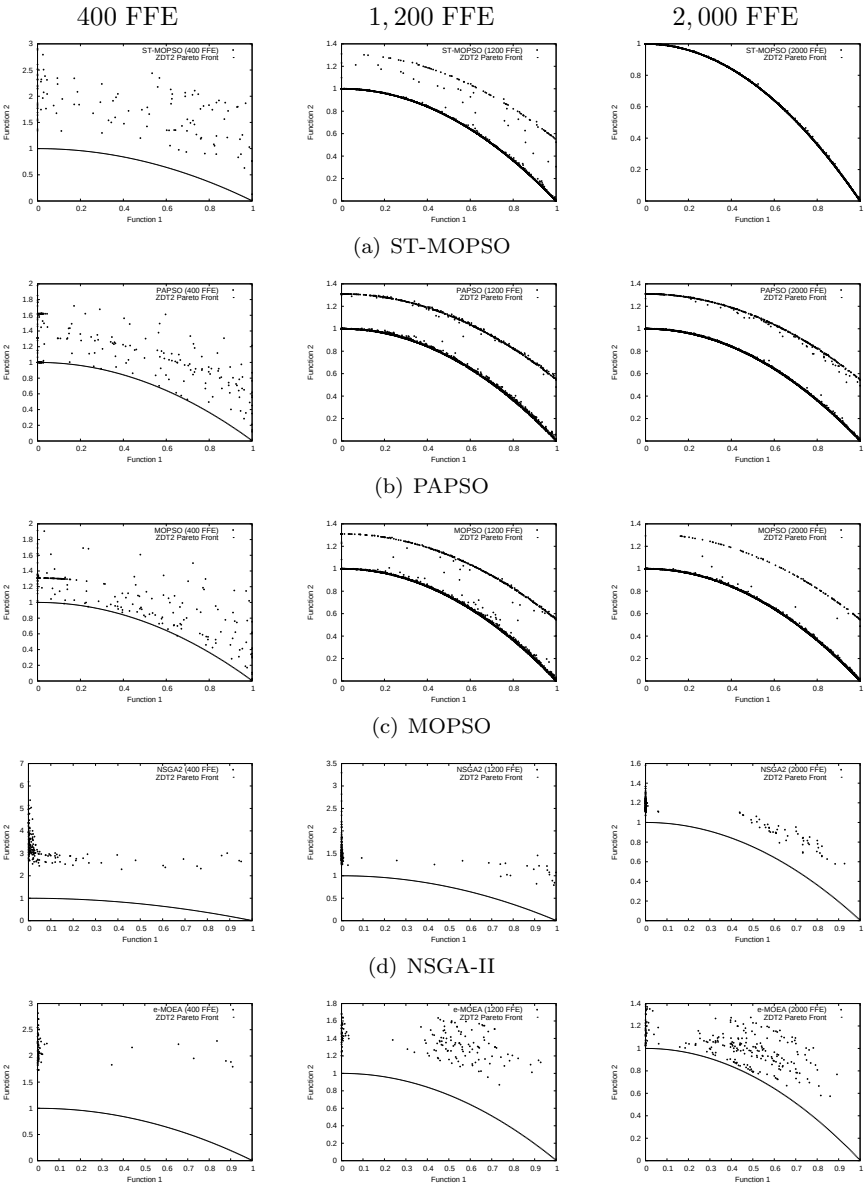


(b) SC



(c) Number of points

Figure 10: Box-plots produced from the results of 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the ZDT1 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) ST-MOPSO

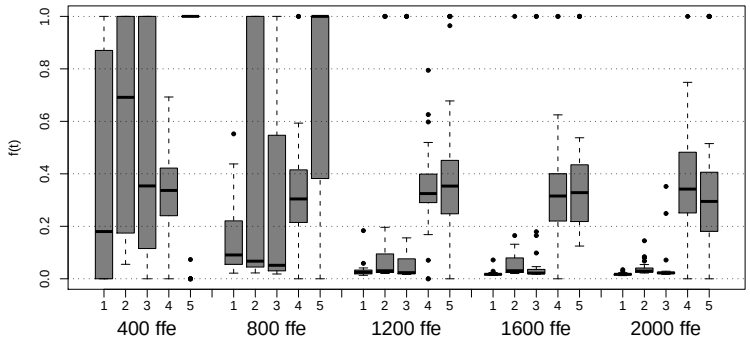
(b) PAPSO

(c) MOPSO

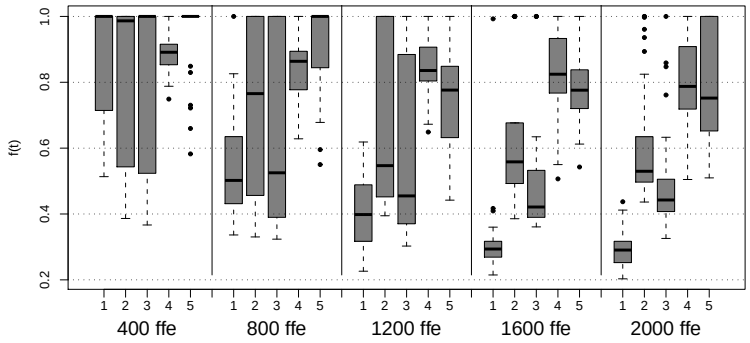
(d) NSGA-II

(e) ε-MOEA

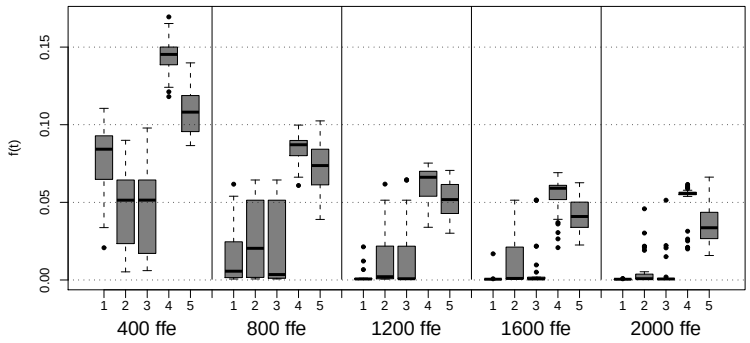
Figure 11: Solutions produced by 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the ZDT2 test function (using 400, 1200 and 2000 fitness function evaluations).



(a) SDC

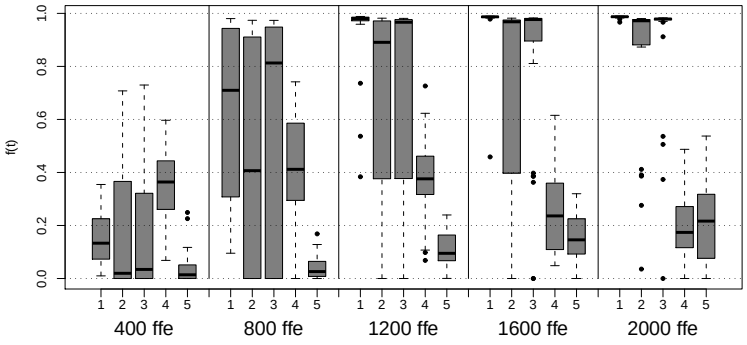


(b) Spread

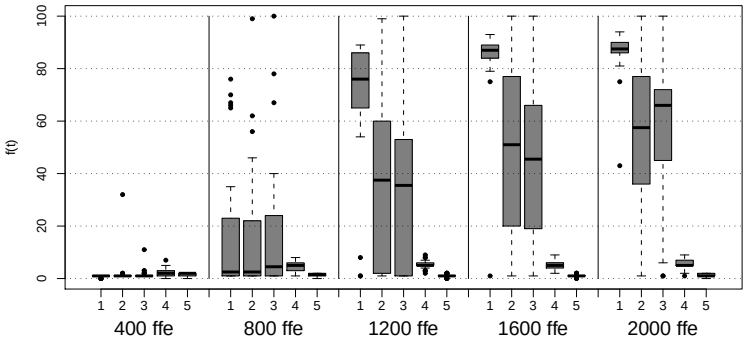


(c) IGD

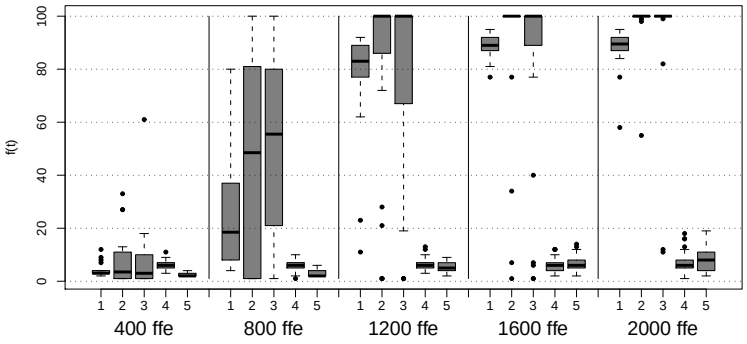
Figure 12: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the ZDT2 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) HVR

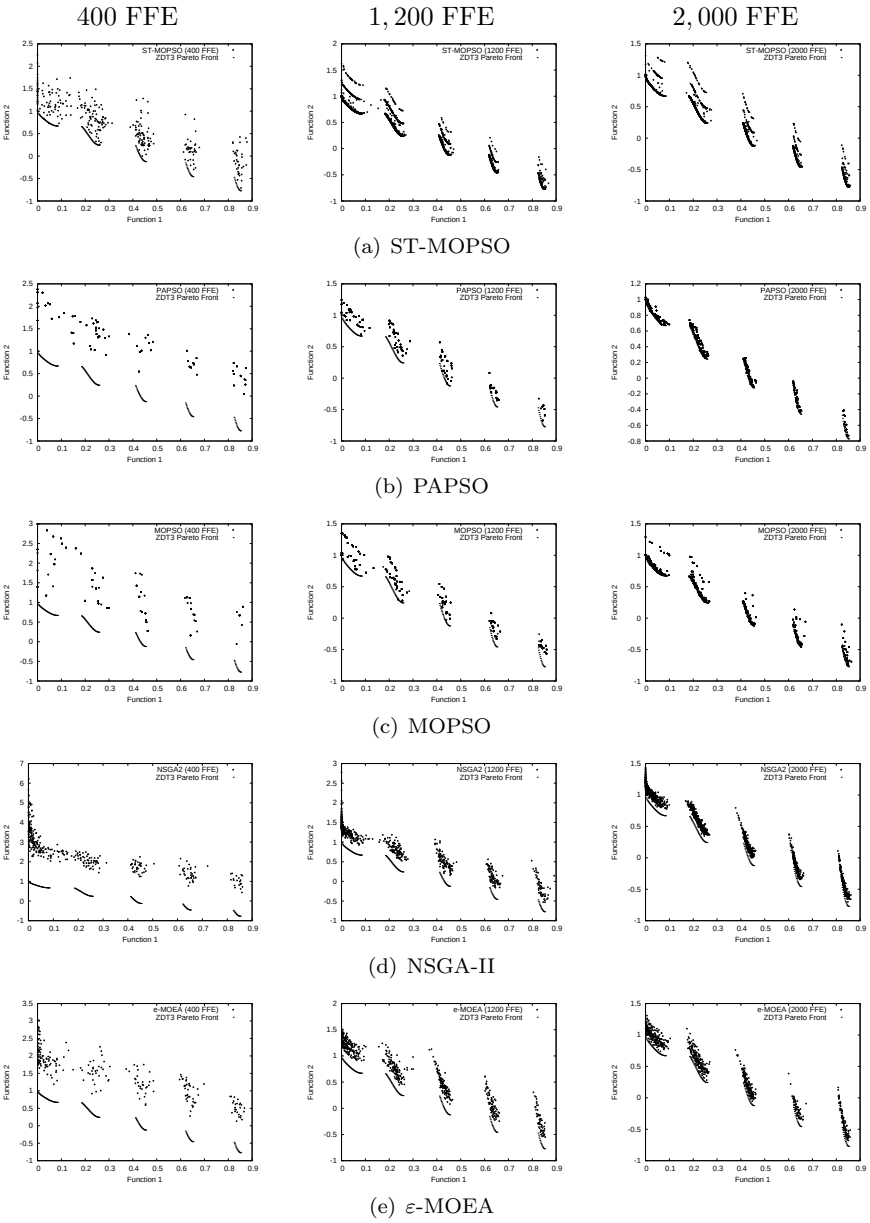


(b) SC



(c) Number of points

Figure 13: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the ZDT2 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) ST-MOPSO

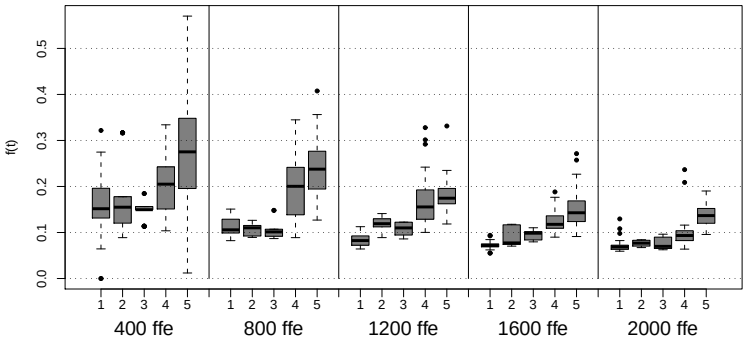
(b) PAPSO

(c) MOPSO

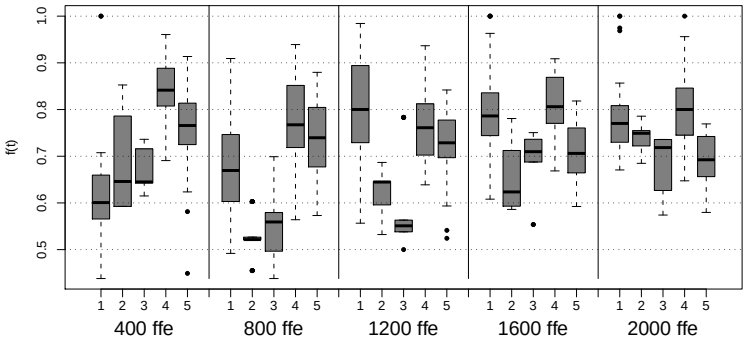
(d) NSGA-II

(e) ε-MOEA

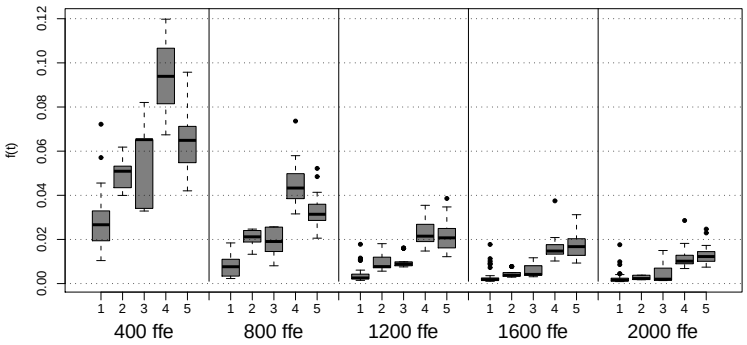
Figure 14: Solutions produced by 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the ZDT3 test function (using 400, 1200 and 2000 fitness function evaluations).



(a) SDC

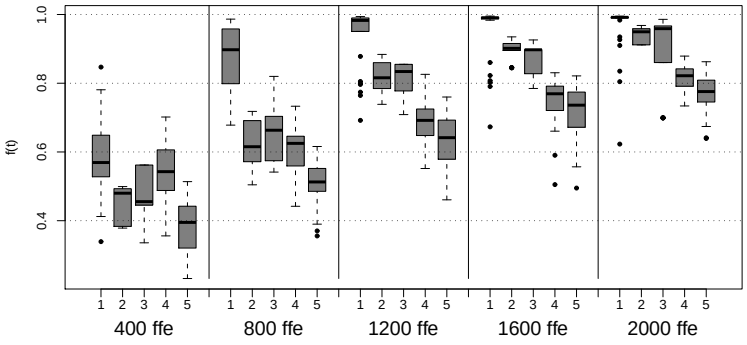


(b) Spread

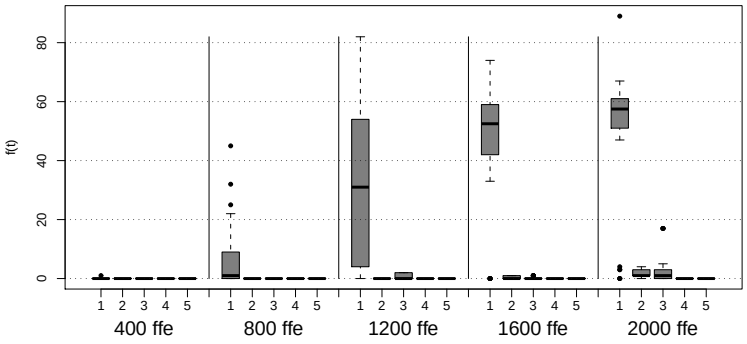


(c) IGD

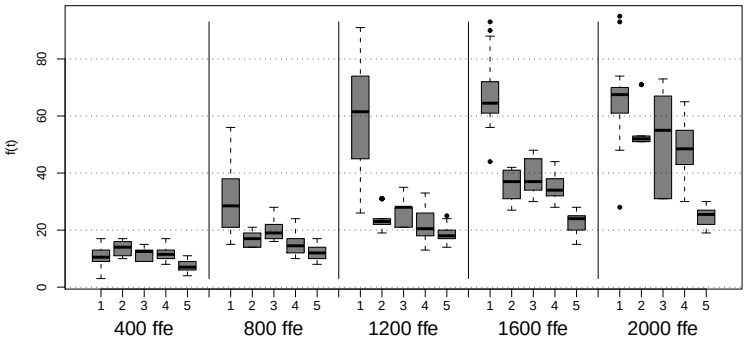
Figure 15: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the ZDT3 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) HVR

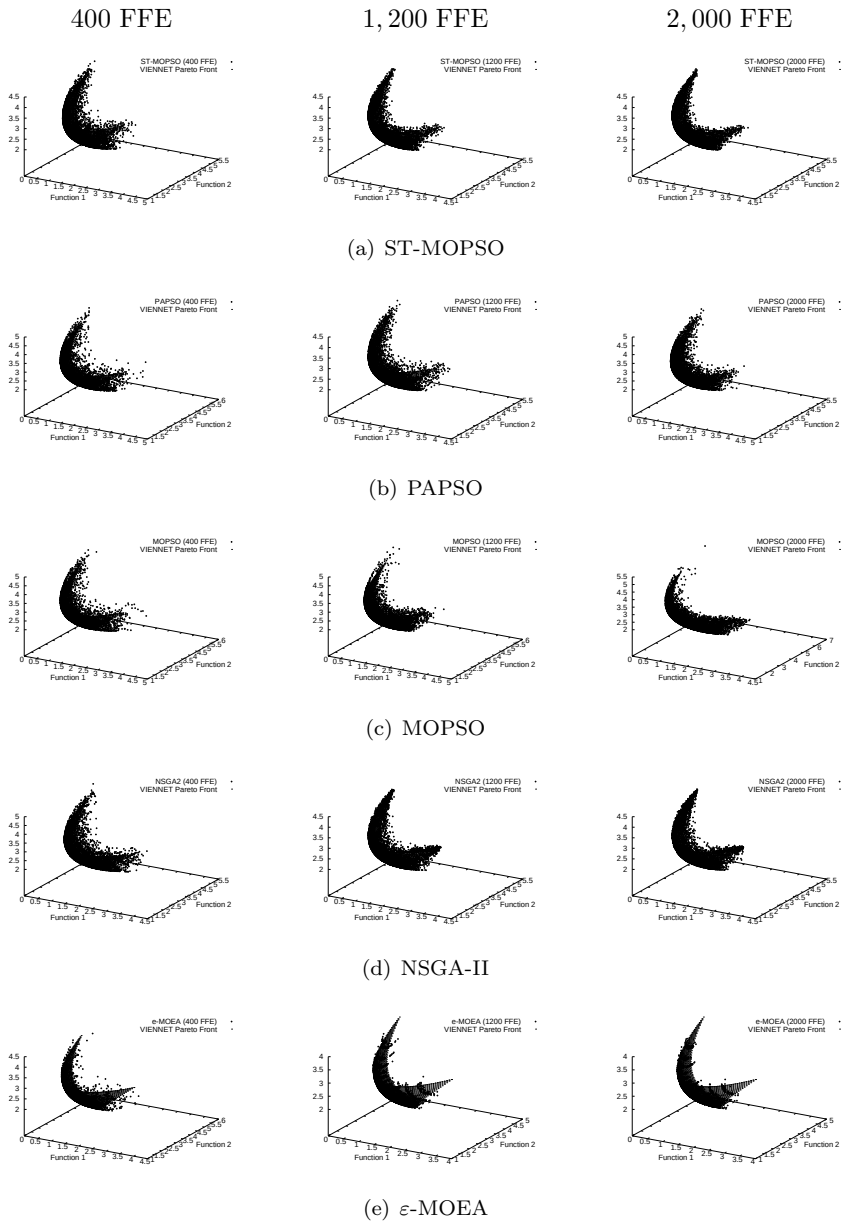


(b) SC



(c) Number of points

Figure 16: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the ZDT3 test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) ST-MOPSO

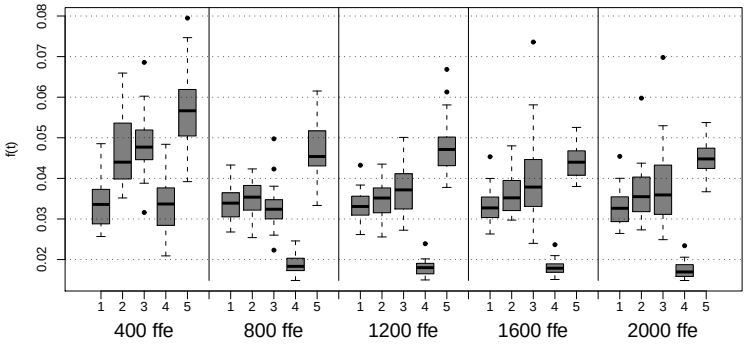
(b) PAPSO

(c) MOPSO

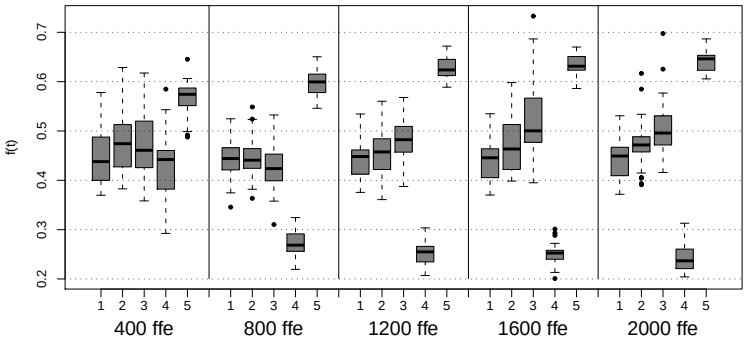
(d) NSGA-II

(e) ε-MOEA

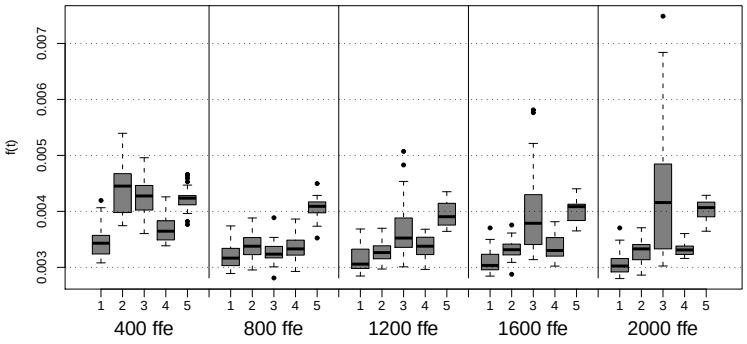
Figure 17: Solutions produced by 1) ST-MOPSO, 2) PAPSO 3) MOPSO 4) NSGA-II, 5) ε-MOEA algorithms in 30 independent runs in the VIENNET test function (using 400, 1200 and 2000 fitness function evaluations).



(a) SDC

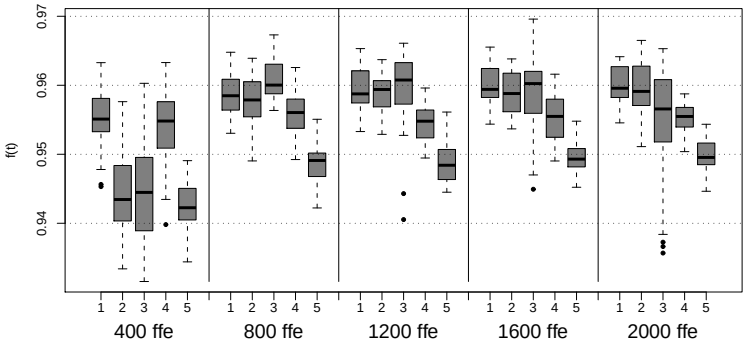


(b) Spread

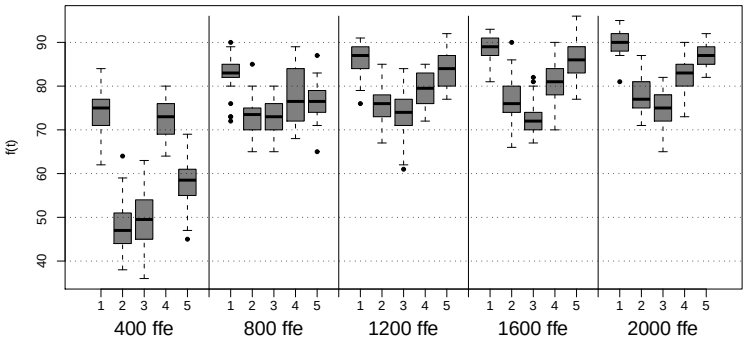


(c) IGD

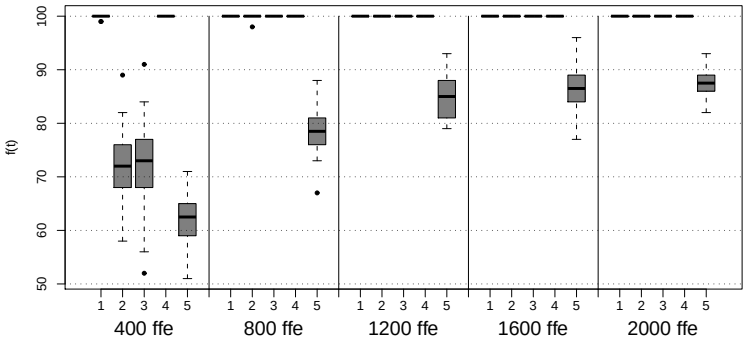
Figure 18: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the VIENNET test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).



(a) HVR



(b) SC



(c) Number of points

Figure 19: Box-plots produced from the results of 1) MOPSO, 2) NSGA-II, 3) ε-MOEA in 30 independent runs in the VIENNET test function (using 400, 800, 1200, 1600 and 2000 fitness function evaluations).