A Painless Gradient-assisted Multi-objective Memetic Mechanism for Solving Continuous Bi-objective Optimization Problems

Adriana Lara López Carlos A. Coello Coello Oliver Schütze

***Abstract*— In this work we present a simple way to introduce gradient-based information as a means to improve the search performed by a multi-objective evolutionary algorithm (MOEA). Our proposal can be easily incorporated into any MOEA, and is able to improve its performance when solving continuous bi-objective problems. We propose a novel mechanism to control the balance between the local search, and the global search performed by a MOEA. We discuss the advantages of the proposed method and its possible use when dealing with more objectives. Finally, we provide some guidelines regarding the use of our proposed approach.**

I. Introduction

A Multi-objective Optimization Problem (MOP) is defined as

$$\begin{aligned} &\text{Minimize} \quad \{f_1(x), f_2(x), \ldots, f_m(x)\} \\ &\text{subject to} \quad x \in S \end{aligned} \tag{1}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$ are known as the objective functions of the problem. The solution set of this problem is based on the Pareto dominance relation [1], and is known as the *Pareto optimal set*. The image of the Pareto optimal set is called the *Pareto front* of the problem.

Introducing gradient information of a MOP, when available, into a Multi-objective Evolutionary Algorithm (MOEA) is a research topic that has attracted a lot of interest in recent years. We identify some advantages of this sort of coupling as the following: First, it can accelerate convergence towards the Pareto front [2], [3], [4], [5]. Second, it can produce more accurate solutions [3]. A possible third advantage is that the use of gradient information could be useful to deal with the so-called "dominance resistant solutions" [6], but this idea has not been properly explored yet, to the authors' best knowledge. Here, we will be focusing only on the first two advantages previously indicated.

There are several hybrids of MOEAs and gradient-based methods currently available in the specialized literature [5], [7], [8], [3], [9]. Such approaches normally replace [5] or add [3], [9] existing evolutionary operators such that the available gradient information is used to guide the search.

Gradient information provides a MOEA with search directions that make it perform more accurate movements. However, computing such search directions is also a multi-objective problem since each objective provides its own (gradient-based) search direction; therefore, all of these directions need to be properly combined into a single one, in order to perform a line search procedure —particularly adapted for the multiobjective case. First results using these ideas are presented in [10], [11], [12] and [2].

Another major issue when incorporating gradient information into a MOEA is how to provide a proper balance between the local search (i.e., using the gradient information) and the global search (i.e., using the MOEA) [13]. In fact, such a balance is problem-dependent. For example, in [7] the use of the local search is proposed after the evolutionary process, and the interleaving between the local and the global search takes place after a certain (fixed) number of objective function evaluations. However, other proposals use the gradient information to overtake solutions towards the Pareto optimal set, and adopt the evolutionary search in a second stage (see for example [8], [4]). In any case, none of these types of strategy are really ideal, since the best would be to have an adaptive mechanism that allows the two types of search to interleave during the run of the MOEA, such that each of them intervenes whenever needed.

The aim of the work reported here is precisely to propose an adaptive mechanism that allows the local and global search strategies to interleave. In our proposed approach, based on the observation of the cones conformed by the gradients of the functions, the algorithm can automatically assign more or less resources either to the local or to the global search procedure, as deemed necessary.

We will focus on the solution of unconstrained bi-objective optimization problems, since only two gradients can effortless be combined into a common descent direction. However, the control part of our proposed approach could be used for problems with a higher number of objectives.

The remainder of this paper is organized as follows. In the next section, we provide the basic theoretical background required to understand the rest of the paper. We also provide a mathematical justification for the effectiveness of our proposal. In Section III, we describe the proposed memetic MOEA (i.e., a global search engine hybridized with a local search mechanism that is based on the use of gradient information). Then, we present some experimental results on several test problems. A discussion of some of the theoretical and practical aspects of our proposed approach is provided in Section V. Finally, our conclusions, as well as some potential

The authors are with the Departamento de Computación, CINVESTAV-IPN, Av. IPN No. 2508, Col. San Pedro Zacatenco, México, D.F. 07360, MEXICO (email: alara@computacion.cs.cinvestav.mx, {ccoello, schuetze}@cs.cinvestav.mx). The second author is also with the UMI-LAFMIA 3175 CNRS at CINVESTAV-IPN.

paths for future research are provided in Section VI.

II. Basic Background

A. Descent Directions and Cones

Let $f_1, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$, $f_i \in C^1$ and $\langle \cdot, \cdot \rangle$ denotes the standard inner product in $\mathbb{R}^n$. We assume that $m \leq n$, and let $\nabla f_i(x)$ be the gradient of the function f_i in x.

Definition 2.1: For each $i \in \{1, \ldots, m\}$ we define:

$$H_{x,i} = \left\{ v \in \mathbb{R}^n \ : \left\langle \frac{\nabla f_i(x)}{||\nabla f_i(x)||}, v \right\rangle = 0 \right\}$$

$$H^+_{x,i} = \left\{ v \in \mathbb{R}^n \ : \left\langle \frac{\nabla f_i(x)}{||\nabla f_i(x)||}, v \right\rangle \geq 0 \right\}$$

$$H^-_{x,i} = \left\{ v \in \mathbb{R}^n \ : \left\langle \frac{\nabla f_i(x)}{||\nabla f_i(x)||}, v \right\rangle \leq 0 \right\}$$

and call it the *Descent Cone* of x to the set

$$C_x(-, -, \ldots, -) = \cap_{i=1}^m H^-_{x,i.}$$

Definition 2.2: A vector $\nu \in \mathbb{R}^n$ is called a *descent direction* of the point $x \in \mathbb{R}^n$ if

$$\nu \in C_x(-, -, \ldots, -).$$

In other words, a descent direction is such that the directional derivatives with respect to ν in x are non positive, *i.e.* $\langle \nabla f_i(x), \nu \rangle \leq 0$ for all $i \in 1, \ldots, m$. This means that if we perform a small movement over ν, we obtain a local improvement (decrease) simultaneously for all the objective functions.

B. The Computed Direction

We base the computation of the descent direction on the next Proposition. This fact has been already observed (e.g. [14]) but has not been exploited in memetic algorithms yet.

Proposition 2.1: Let $x \in \mathbb{R}^n$, and $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ define a bi-objective MOP. Then, the direction

$$\overline{\nabla}_x = -\left(\frac{\nabla f_1(x)}{||\nabla f_1(x)||} + \frac{\nabla f_2(x)}{||\nabla f_2(x)||} \right), \tag{2}$$

where $||\cdot|| = ||\cdot||_2$, is a descent direction at x for the MOP.

Proof: Let us denote $\nabla_i := \frac{\nabla f_i(x)}{||\nabla f_i(x)||}$ for $i = \{1, 2\}$, and θ be the angle between ∇_1 and ∇_2. Then

$$\begin{aligned} \langle \overline{\nabla}_x, \nabla_1 \rangle &= \langle -(\nabla_1 + \nabla_2), \nabla_1 \rangle \\ &= -1 \left(\langle \nabla_1, \nabla_1 \rangle + \langle \nabla_2, \nabla_1 \rangle \right) \\ &= -1 \left(1 + ||\nabla_2|| \, ||\nabla_1|| \cos(\theta) \right) \\ &= -1 - \cos(\theta) \\ &\leq 0. \end{aligned} \tag{3}$$

Similarly, $\langle \overline{\nabla}_x, \nabla_2 \rangle \leq 0$; then, $\overline{\nabla}_x$ is a descent direction of the point x for the defined MOP. □

Unfortunately, Proposition 2.1 cannot be generalized for more than two objective functions. Nevertheless, the remainding parts of this proposal can still be used in the general multi-objective case, by just applying other approaches (see Section V) to obtain the descent direction.

III. Our Proposed Algorithm

Next, we present in Algorithm 1 our version of a MOEA hybridized with gradient information. To show our proposed coupling, we chose the widely used NSGA-II [15] as our global search engine. Nevertheless, the coupling with other MOEAs is also possible as will be later discussed.

Algorithm 1 Hybrid GH-NSGA

```
1:  procedure GH-NSGA(N,G)
2:      Generate a Random Population P.
3:      Evaluate Objective Function Values.
4:      Fast Non-Dominated Sort
5:      Crowding Distance Assignment
6:      for i ← 1, ..., G do
7:          Generate Offspring Population P_offs
8:          Set P ← P ∪ P_offs:
9:          Fast Non-Dominated Sort
10:         Crowding Distance Assignment
11:         LocalSearch(i, P)
12:     end for
13: end procedure

14: procedure LocalSearch(i, P)
15:     if i ≡ 0 (mod k) then
16:         Compute e as in equation (4).
17:         Form the set E taking e individuals
18:         randomly selected out of R_1(P).
19:         for all a ∈ E do
20:             if local improvement is possible then
21:                 Apply line search to obtain a'.
22:                 Replace a ← a'.
23:                 Set a' ∈ R_1(P).
24:                 Set the crowding distance of a' as ∞.
25:             end if
26:         end for
27:     end if
28: end procedure
```

The procedures "Fast Non-Dominated Sort", "Crowding Distance Assignment" and "Generate Offspring Population" are well-known components of the NSGA-II. Their details can be found in [15]. The parameters N and G in Algorithm 1, represent the population size and the maximum number of generations, respectively.

Algorithm 1 places the local search inside the NSGA-II just after the reproduction and the ranking-crowding process. The local search is applied only to non-dominated individuals, but not to all of them. Our control mechanism (global vs. local) uses information that has already emerged from the original ranking process.

One of the main issues when using gradient-based tools is how to balance the computational cost of the method with the improvements achieved. The currently available MOEAs that use descent directions as local search engines have two sources of computational cost: the first is associated to the fitness function evaluations required to estimate the gradients

and to perform the line search. The second source is related to the computation of the descent direction itself. In this sense, and unlike previous approaches, our proposal has the advantage of having a zero cost for the computation of the descent directions. We claim that this procedure is the simplest way to combine two functions gradients and that it can not be generalized to more than two functions since the arithmetic combination of them does not produce descent directions in general.

A. Line Search

To apply the line search (line 21 of Algorithm 1) we calculate the descent direction $\overline{\nabla}_x$ using equation (2), and we obtain the individual x' as

$$x' = x + t_x \overline{\nabla}_x.$$

To estimate t_x, we use an Armijo-like rule starting from a size t_{max}, and we reduce $t \leftarrow t/2$ at each iteration, until $f_i(x') \leq f_i(x)\ \forall i$ (or the Armijo condition) is fulfilled. The procedure is sensitive to this t_{max} parameter as any other line-search procedure is to the initial step length choice.

B. Stopping Criterion

Since it could happen that a certain point x is too close to a Pareto optimum point, we propose to set a stopping criterion (line 20 of Algorithm 1) to apply the steepest descent. This criterion is related to certain small tolerance $0 < \epsilon_{tol} < 0.01$ and is given by the next rule: if

$$\langle \nabla_1, \nabla_2 \rangle < -1 + \epsilon_{tol}$$

holds, then, no local search movement is performed for the point x. This stopping criterion is inspired on the Karush-Kuhn-Tucker (KKT) optimality conditions [16].

C. Balance Control

In order to have an adaptive strategy, we introduced a control mechanism that incorporates the local search based on the number of non-dominated elements available at each iteration (we discuss this in more detail in Section V).

To calculate the control parameter e we propose the next rule:

$$e = \begin{cases} 0 & \text{if} \quad |R_1(P)| < (0.1)(N) \\ \left\lfloor \frac{|R_1(P)|}{(0.1)(N)} \right\rfloor & \text{otherwise.} \end{cases} \tag{4}$$

where $|R_1(P)|$ is the cardinality of the subset $R_1(P)$ (lines 18 and 23 of Algorithm 1) formed by the elements in P whose rank is one. N is the population size and $\lfloor \cdot \rfloor$ represents the *floor* function.

The idea behind the parameter k (line 15[1] of Algorithm 1) is that the local search is improving only a few individuals at a certain moment and, since we have a population-based approach (the MOEA acting as our global search engine) we should let the evolution do its job. In other words, we are expecting that most of the effort will come from the global procedure, and that the gradient-based local search will only refine the solutions generated by the global engine. Not applying the local search at each iteration will help us to make its cost more affordable. At the same time, this will allow us to push the population of the MOEA towards better solutions. Thus, parameter k determines how often do we apply the local search. We believe that this parameter must be varied adaptively during the search as a second control mechanism for resources balance.

[1] $a \equiv b$ (mod m) is that both numbers a and b leave the same residue when divided by m.

IV. Test problems and results

For assessing the performance of our proposed approach, and in order to get unconstrained problems, we used the modified Zitzler-Deb-Thiele (ZDT) problems presented in [5] and [17] (with the difference that we do not need the twice differentiability property). These test functions are defined in Table I (ZDT5 was not included because it is a binary problem).

TABLE I
Test Problems Adopted

Problem	Functions	Domain
ZDT 1	$f_1(x) = x_1$ $f_2(x) = g(x)(2 - \sqrt{f_1(x)/g(x)})$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0,1] \times [-1,1]^n$
ZDT 2	$f_1(x) = x_1$ $f_2(x) = g(x)(2 - (f_1(x)/g(x))^2)$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0,1] \times [-1,1]^n$
ZDT 3	$f_1(x) = x_1$ $f_2(x) = g(x)(2 - \sqrt{f_1(x)/g(x)} -$ $(f_1(x)/g(x))\sin(10\pi f_1))$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0,1] \times [-1,1]^n$
ZDT 4	$f_1(x) = x_1$ $f_2(x) = g(x)(2 - \sqrt{f_1(x)/g(x)})$ $g(x) = 1 + 10(n-1) +$ $\sum_{i=2}^{n}(x_i^2 - 10\cos 4\pi f_1)$	$[0,1] \times [-5,5]^n$
ZDT 6	$f_1(x) = 1 - \exp -4x_1$ $f_2(x) = g(x)(2 - (f_1(x)/g(x))^2)$ $g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i^2$	$[0,1] \times [-1,1]^n$

TABLE II
Number of decision variables used for each test problem

Problem	Original Size	Extended Size
ZDT 1	30 variables	60 variables
ZDT 2	30 variables	60 variables
ZDT 3	30 variables	60 variables
ZDT 4	10 variables	15 variables
ZDT 6	10 variables	15 variables

All the experiments were coded using C language. We took the implementation of NSGA-II available from its author at `http://www.iitk.ac.in/kangal/codes.shtml`. The parameters used in our experiments were the following: Population size $N = 100$, $\epsilon_{tol} = 0.0001$, $k = 2$ (but similar results were obtained for values under 10) and $t_{max} = 2$.

Regarding the computational costs involved in practice, we are assuming that the user estimates the gradients of the objective functions using an evaluations-saver method

such as Automatic Differentiation [18]. This sort of approach introduces significant savings in the computational cost of the gradient values.

It is worth noting that we adopted the previously described test problems both, with their original dimensionality and with a higher number of decision variables. The specific dimensionalities adopted are indicated in Table II.

TABLE III
COMPARISON OF RESULTS USING THE SET COVERAGE INDICATOR FOR THE ORIGINAL VERSIONS OF THE TEST PROBLEMS.

Problem	NSGA-II version:	Set Coverage 5,000 f. evals. Mean	σ	10,000 f. evals. Mean	σ
ZDT 1	Hybrid $<$ Plain	0.98	0.03	0.78	0.08
	Plain $<$ Hybrid	0.00	0.00	0.00	0.00
ZDT 2	Hybrid $<$ Plain	0.91	0.21	0.88	0.08
	Plain $<$ Hybrid	0.01	0.07	0.00	0.00
ZDT 3	Hybrid $<$ Plain	0.90	0.11	0.60	0.09
	Plain $<$ Hybrid	0.03	0.06	0.01	0.03
		20,000 f. evals. Mean	σ	40,000 f. evals Mean	σ
ZDT 4	Hybrid $<$ Plain	0.66	0.29	0.32	0.19
	Plain $<$ Hybrid	0.11	0.27	0.00	0.00
ZDT 6	Hybrid $<$ Plain	1.00	0.00	1.00	0.00
	Plain $<$ Hybrid	0.00	0.00	0.00	0.00

TABLE IV
COMPARISON OF RESULTS USING THE SET COVERAGE INDICATOR, FOR THE EXTENDED VERSIONS OF THE TEST PROBLEMS.

Problem	NSGA-II version:	Set Coverage 5,000 f. evals. Mean	σ	10,000 f. evals. Mean	σ
ZDT 1	Hybrid $<$ Plain	0.99	0.02	1.00	0.01
	Plain $<$ Hybrid	0.00	0.00	0.00	0.00
ZDT 2	Hybrid $<$ Plain	0.86	0.30	0.90	0.24
	Plain $<$ Hybrid	0.05	0.17	0.00	0.00
ZDT 3	Hybrid $<$ Plain	0.88	0.17	0.95	0.05
	Plain $<$ Hybrid	0.10	0.18	0.01	0.03
		20,000 f. evals. Mean	σ	40,000 f. evals Mean	σ
ZDT 4	Hybrid $<$ Plain	0.55	0.49	0.70	0.29
	Plain $<$ Hybrid	0.52	0.48	0.13	0.32
ZDT 6	Hybrid $<$ Plain	1.00	0.00	1.00	0.00
	Plain $<$ Hybrid	0.00	0.00	0.00	0.00

We compared the plain NSGA-II with respect to our proposed hybrid approach. All the experiments were run until reaching a certain number of function evaluations (5,000 for ZDT1, ZDT2 and ZDT3; 20,000 for ZDT4 and ZDT6) which corresponds to the moment when certain reasonable proximity to the front has been reached (see the right handside plots of Figures 6 to 10). Then, we allowed the algorithms to perform twice these numbers of evaluations. It is worth noting that the same number of evaluations were adopted for the two cases: for the original test problems and for their extended versions.

Tables III to VI show the mean and the standard deviations over 30 independent runs, regarding the indicators *Set Coverage* and *Inverted Generational Distance* (IGD), as defined in [19]. The calculation of *IGD* was done using the parameter $p = 1$.

Table III shows that, for the first number of evaluations, the hybrid approach almost always set-covers the plain MOEA (its values are close to one), while the plain NSGA-II almost never set-covers the hybrid approach (its values are close to zero). The only exceptions are ZDT6, where the values of the coverage are exactly one and zero (which means total coverage of the hybrid over the original approach and never the opposite) and ZDT4 in which the coverage difference was reduced. When performing twice the number of evaluations, the plain NSGA-II was able to converge closer to the true Pareto front, but was never able to set-cover the hybrid approach.

In Table IV, we show the results obtained for the extended versions of the test problems. We can see here similar results to those obtained when dealing with the original versions of the test problems. However, in this case the outperformance of the hybrid over the plain NSGA-II improved as the number of iterations was increased. Tables V and VI support these results, and show the IGD which measures both, proximity to the front and spread of solutions.

It is worth noticing that, since ZDT4 is a highly multifrontal MOP, then, a gradient-based method is expected to get stuck when attempting to solve it, producing, as a consequence, a negative impact on the performance of the hybrid MOEA. However, our results do not indicate that this is always the case, but this and ZDT2 were the least stable in terms of the standard deviation for the IGD indicator.

Since the two algorithms compared use the same crowding selection procedure, the distance towards the Pareto front does not decrease monotonically at the last stages of the search, since some good solutions could be deleted at further iterations. This is also the reason why no useful information (for comparison purposes) can be obtained from letting the algorithms to perform a higher number of objective function evaluations.

Figures 1 to 5 show the Pareto fronts generated for each of the test problems in their two instances (original and extended), and for the two numbers of iterations adopted. These Pareto fronts correspond to randomly selected runs.

Figures 6 to 10 show the average over 30 runs for the number of non-dominated points—the basis of our control mechanism—and the *Generational Distance* (GD) [19] as the number of function evaluations increases. These comparisons are presented in the original and also in the extended size versions of the problems.

V. GENERAL DISCUSSION

When hybridizing MOEAs with gradient-based procedures, an obvious question that arises is if this sort of hybrid scheme is more cost-effective than the use of a plain MOEA. This cannot be easily answered, and few studies that look into this are currently available. However, it is known that the answer to this question depends on two things: the specific features of the problem to be solved, and the effectiveness of the mechanism that balances the local search with the

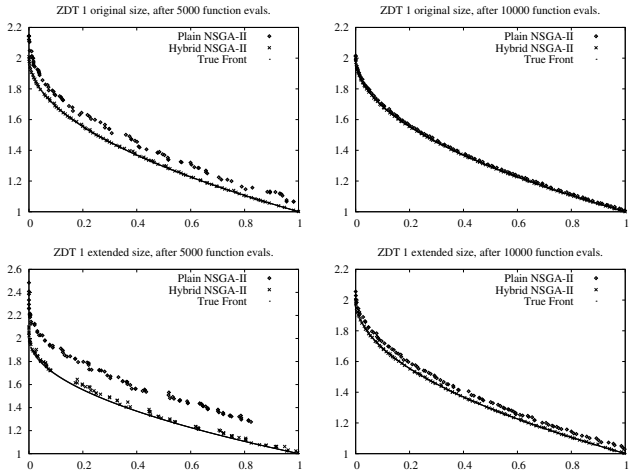

Fig. 1. Pareto fronts corresponding to a random run (seed 0.39) for ZDT1. Original (top) and extended sizes (bottom), 5000 (left) and 10000 (right) evaluations

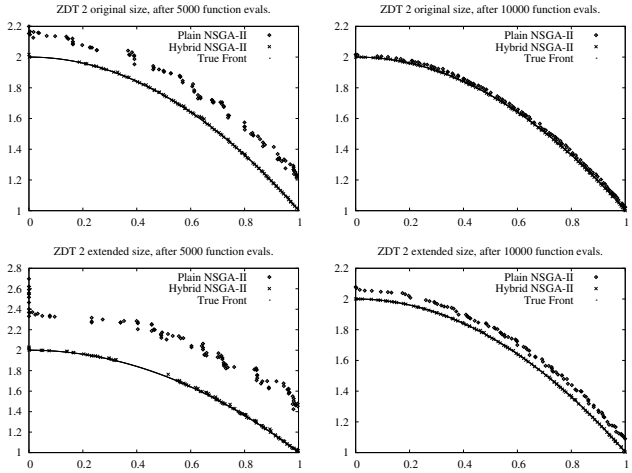

Fig. 2. Pareto fronts corresponding to a random run (seed 0.39) for ZDT2. Original (top) and extended sizes (bottom), 5000 (left) and 10000 (right) evaluations

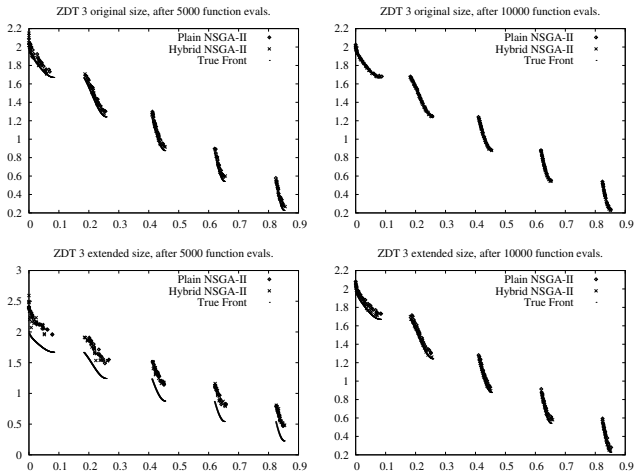

Fig. 3. Pareto fronts corresponding to a random run (seed 0.39) for ZDT3. Original (top) and extended sizes (bottom), 5000 (left) and 10000 (right) evaluations

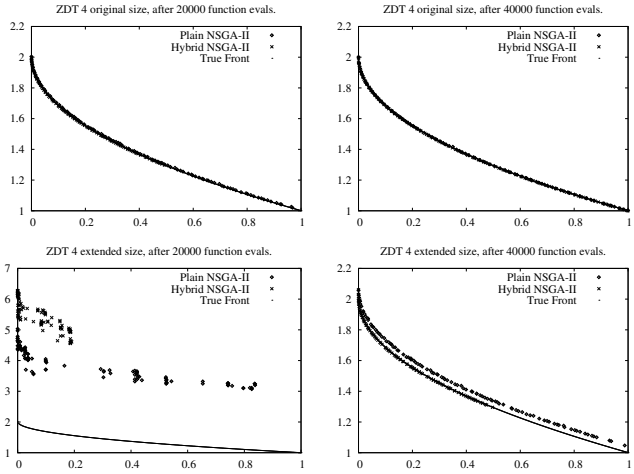

Fig. 4. Pareto fronts corresponding to a random run (seed 0.39) for ZDT4. Original (top) and extended sizes (bottom), 20000 (left) and 40000 (right) evaluations

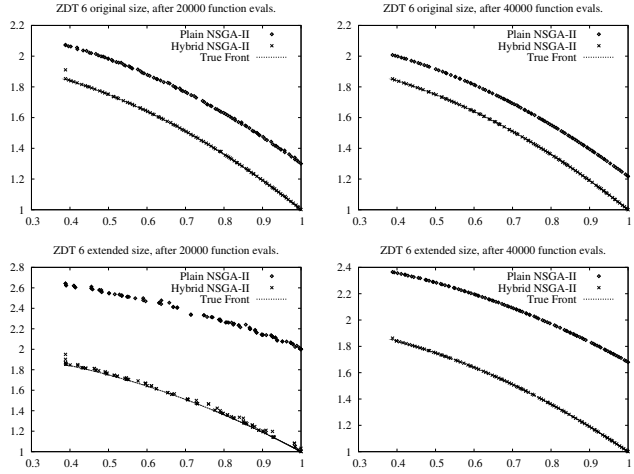

Fig. 5. Pareto fronts corresponding to a random run (seed 0.39) for ZDT4. Original (top) and extended sizes (bottom), 20000 (left) and 40000 (right) evaluations

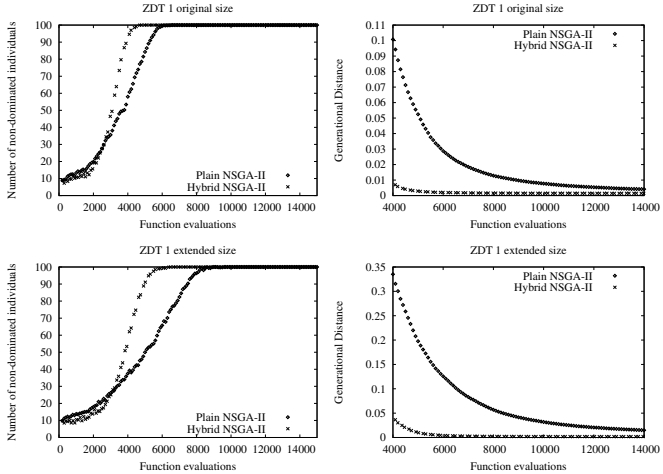

Fig. 6. Average over 30 runs for ZDT1 with both its original size (top) and its extended size (bottom). We also show the number of non-dominated elements (left), and the Generational Distance (right).

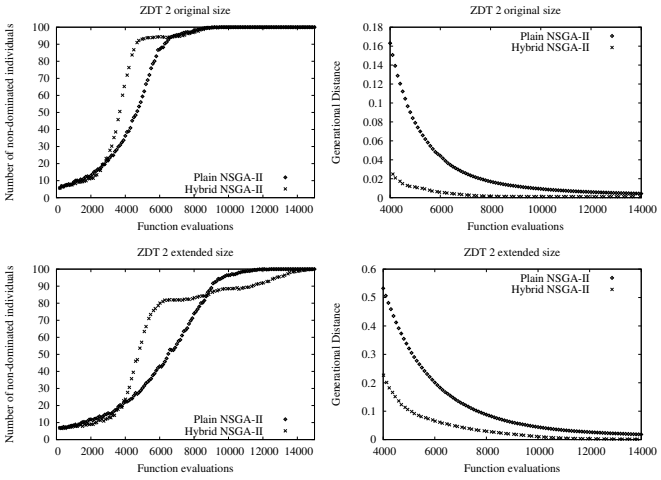

Fig. 7. Average over 30 runs for ZDT2 with both its original size (top) and its extended size (bottom). We also show the number of non-dominated elements (left), and the Generational Distance (right).

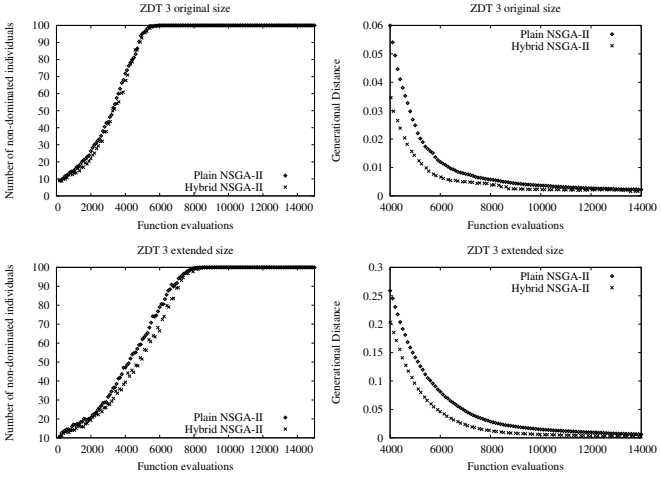

Fig. 8. Average over 30 runs for ZDT3 with both its original size (top) and its extended size (bottom). We also show the number of non-dominated elements (left), and the Generational Distance (right).

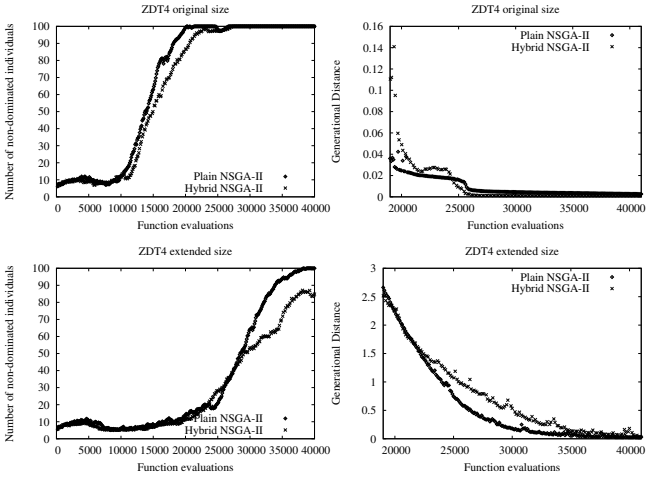

Fig. 9. Average over 30 runs for ZDT4 with both its original size (top) and its extended size (bottom). We also show the number of non-dominated elements (left), and the Generational Distance (right).

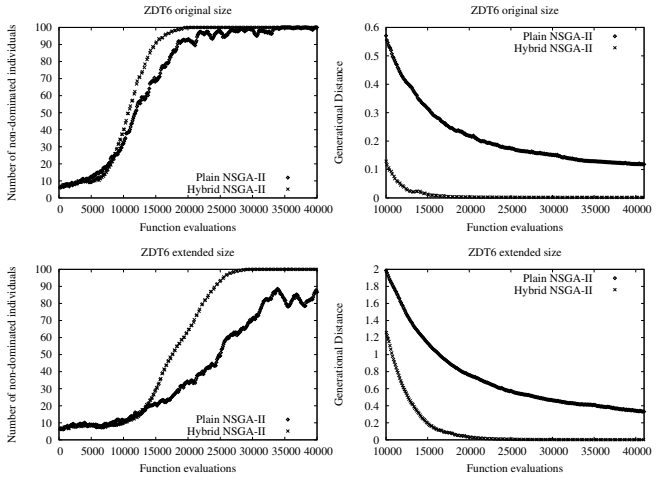

Fig. 10. Average over 30 runs for ZDT6 with both its original size (top) and its extended size (bottom). We also show the number of non-dominated elements (left), and the Generational Distance (right).

TABLE V

COMPARISON OF RESULTS FOR THE IGD INDICATOR, ADOPTING THE TEST PROBLEMS WITH THEIR ORIGINAL SIZES.

Problem	NSGA-II version:	IGD 5,000 function evals. Mean	5,000 function evals. Std. Dev.	10,000 function evals. Mean	10,000 function evals. Std. Dev.
ZDT 1	Hybrid	9.059E-03	9.080E-03	4.798E-03	5.318E-04
	Plain	5.134E-02	1.129E-02	9.349E-03	8.545E-04
ZDT 2	Hybrid	4.437E-02	1.109E-01	7.108E-03	1.264E-02
	Plain	1.114E-01	1.396E-01	1.446E-02	1.398E-02
ZDT 3	Hybrid	1.959E-02	1.252E-02	6.084E-03	4.021E-03
	Plain	3.722E-02	9.161E-03	7.443E-03	6.264E-04
		20,000 function evals. Mean	20,000 function evals. Std. Dev.	40,000 function evals Mean	40,000 function evals Std. Dev.
ZDT 4	Hybrid	1.237E-01	1.233E-01	4.511E-03	1.441E-04
	Plain	6.191E-02	8.149E-02	5.483E-03	9.147E-04
ZDT 6	Hybrid	2.859E-03	3.342E-04	2.422E-03	1.828E-04
	Plain	3.262E-01	7.280E-02	1.806E-01	3.616E-02

TABLE VI

COMPARISON OF RESULTS FOR THE IGD INDICATOR, ADOPTING THE TEST PROBLEMS IN THEIR EXTENDED VERSIONS.

Problem	NSGA-II version:	IGD 5,000 function evals. Mean	5,000 function evals. Std. Dev.	10,000 function evals. Mean	10,000 function evals. Std. Dev.
ZDT 1	Hybrid	1.269E-02	5.371E-03	4.816E-03	1.682E-04
	Plain	1.899E-01	3.366E-02	3.273E-02	5.349E-03
ZDT 2	Hybrid	1.785E-01	3.038E-01	4.189E-02	1.016E-01
	Plain	3.167E-01	1.102E-01	5.197E-02	4.667E-02
ZDT 3	Hybrid	1.086E-01	2.525E-02	1.206E-02	7.814E-03
	Plain	1.487E-01	2.186E-02	2.632E-02	9.028E-03
		20,000 function evals. Mean	20,000 function evals. Std. Dev.	40,000 function evals Mean	40,000 function evals Std. Dev.
ZDT 4	Hybrid	2.018E+00	9.879E-01	1.137E-01	9.409E-02
	Plain	2.086E+00	8.390E-01	3.600E-02	2.234E-02
ZDT 6	Hybrid	1.080E-02	1.044E-02	2.439E-03	1.434E-04
	Plain	1.023E+00	1.293E-01	4.985E-01	9.087E-02

global search. Excluding the computation of the descent direction (Proposition 2.1), the ideas presented in this paper can be used to hybridize MOEAs to deal with more than two objectives. For those cases, we suggest to use either the procedure proposed by Fliege et al. [10] or the one by Schäffler et al. [11]. It is worth noting that, in this case, it is necessary to use a solver for convex quadratic optimization or, at least, a linear optimization solver.

A. Adaptive Switch

One observation taken from [20] is that the descent cone is wider for farther points than for those close to KKT points. In particular, the descent cone shrinks down while the search gets closer to the optimum. Thus, the probability to locally improve the point x with any kind of perturbation is high when x is far, being this the case during the first stages of the evolutionary search, when many points are likely to be replaced by non-dominated ones. On the other hand, when a point highly improves, it is unlikely that new points, stochastically generated, could replace it—since the descent cone shrinks down—and it will remain in this state for several iterations. This is the case when a trusty movement towards a descent direction is more necessary, and, therefore, the use of local search becomes cost-effective. However, since different MOPs have different fitness landscapes, we do not really know if this state is reached at the end of the search (or at any other stage, for that sake). Thus, it becomes desirable to have an adaptive mechanism that can recognize when the use of local search is useful and when it will produce no gains with respect to the use of the global search engine alone.

Based on this reasoning, we propose a control mechanism based on the cardinality of the set of non-dominated points. The idea is that when the evolutionary search is promising, the local search is scarcely used. On the other hand, when the number of non-dominated solutions grows, our method allows the local search mechanism to be applied more often.

B. Computing The Descent Direction

The advantage of this proposal over other currently available for multi-objective descent directions [11], [10], [2] is that no additional effort is needed for its computation itself. While all the other alternatives need to solve a quadratic or linear optimization problem for each point, we emphasize that for bi-objective problems just an arithmetic operation (a geometrical "average") over the gradients is necessary. This could be attractive for the user because no additional code for solvers has to be added, and the procedure can be directly plugged into the MOEA used.

C. Step Size Control

Once the descent direction ν has been found, a line search must be performed. If a bad step size control is chosen, the local search could consume an extremely high number of function evaluations. The way in which we controlled the step size worked fine for the test problems adopted, but it is advisable that, in general, a limit on the number of allowable function evaluations for the local search strategy is imposed.

Let us point out that the calculation of a suitable step size for the line search in multi-objective optimization is an open problem (in fact, it is a multi-objective problem itself) which clearly deserves more attention.

D. Surviving the Crowding Bound

It can be noticed from Figures 6 to 10 (right handside) that the crowding procedure truncation does not allow the method to converge. In other words, the zero value is never reached when assessing GD for a large number of function evaluations. The worst part is that there is a positive probability to lose, during the selection (when using the crowding procedure), points which have already been improved (by the steepest descent). The alternative that we adopted in this work was to skip the calculation of the crowding value for the elements resulting from the application of the local search (by assigning a special value to them). However, this mechanism only saves these elements for the next generation and it does not prevent that, at future generations, the improved individuals are deleted. Otherwise, we could interfere with the original diversity control process of the MOEA and this could have a negative effect.

As a final suggestion, if guaranteed convergence is searched for the procedure, the observation presented above motivates the study and use of different types of archivers for MOEAs.

VI. Conclusions and Future Work

Most of current multi-objective memetic algorithms have focused on discrete problems (e.g. [13]) in which the local search over neighborhoods is well studied; but for the continuous case, there is not direct comparison with these proposals. On the other hand, gradient-based line search is a powerful tool which implies a high cost but is also a reliable way to produce improvements. This work addresses different aspects related to the hybridization of MOEAs with local search mechanisms based on gradient information. First, we incorporated the gradient of two functions in the most simple possible way, compared with other works which use gradient-based multi-objective line search—since these other procedures require the solution of SQP or linear problems (e.g. [21], [2], [7]). In this way we developed a "plug and play" method that can be easily coupled to many MOEAs with little effort. The hybrid algorithm showed advantages over the plain MOEA in the test problems adopted in their two instances (i.e., with their original dimensionality and with higher dimensionality). The most important advantage of the proposed approach is that no additional quadratic or linear optimization solvers are required to calculate the descent direction for bi-objective problems. This makes our approach cost-free for such types of problems. Our proposal also includes a tolerance controlled stopping criterion in order to avoid applying local search in a particular (almost optimal) point.

Second, we tackled the problem of the balance between local search and global search. This is not a trivial issue, and it has been indeed recognized as one of the main difficulties when designing memetic MOEAs. This work was a first step towards developing a fully adaptive method that can automatically balance the role of each of the two search engines (i.e., the global search and the local search engines) when dealing with continuous problems. We presented a criteria to switch between the local and the global search procedures based on the analysis of the descent cones. The aim of such mechanism is precisely to balance the resources (i.e., function evaluations) assigned to each of the two engines.

As part of our future work, we are interested in coupling alternative archiving methods (see for example [22]) with our approach. Such type of mechanism should be able to preserve good solutions during a longer time, which would be beneficial for the performance of the final algorithm. It is also necessary to develop a deeper investigation about the stability of our proposed balancing mechanism, in order to establish if it is more applicable for general problems than using a fixed probability for the local search application —since this last option is very sensitive to this probability value, and completely dependent of the problem. Finally, assessing and comparing procedures that use evolutionary approaches for the estimation of descent directions (such as [9] and [23]) within multi-objective memetic algorithms is also part of our future work.

ACKNOWLEDGEMENTS

The first author acknowledges support from CONACyT to pursue graduate studies at the Computer Science Department of CINVESTAV-IPN. The second author acknowledges support from CONACyT project no. 103570.

REFERENCES

[1] V. Pareto, *Cours Déconomie Politique*. Lausanne, F. Rouge; Paris, Pichon, 1896.

[2] P. A. Bosman and E. D. de Jong, "Exploiting Gradient Information in Numerical Multi-Objective Evolutionary Optimization," in *2005 Genetic and Evolutionary Computation Conference (GECCO'2005)*, H.-G. B. et al., Ed., vol. 1. New York, USA: ACM Press, June 2005, pp. 755–762.

[3] K. Sindhya, K. Deb, and K. Miettinen, "A Local Search Based Evolutionary Multi-objective Optimization Approach for Fast and Accurate Convergence," in *Parallel Problem Solving from Nature–PPSN X*, G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, Eds. Dortmund, Germ.: Springer. Lecture Notes in Computer Science Vol. 5199, September 2008, pp. 815–824.

[4] A. Lara, C. A. Coello Coello, and O. Schütze, "Using Gradient-Based Information to Deal with Scalability in Multi-objective Evolutionary Algorithms," in *2009 IEEE Congress on Evolutionary Computation (CEC'2009)*. Trondheim, Norway: IEEE Press, May 2009, pp. 16–23.

[5] P. Shukla, "On Gradient Based Local Search Methods in Unconstrained Evolutionary Multi-objective Optimization," in *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. Matshushima, Japan: Springer. Lecture Notes in Computer Science Vol. 4403, March 2007, pp. 96–110.

[6] K. Ikeda, H. Kita, and S. Kobayashi, "Failure of Pareto-based MOEAs: does non-dominated really mean near to optimal?" in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2, 2001.

[7] K. Harada, K. Ikeda, and S. Kobayashi, "Hybridizing of Genetic Algorithm and Local Search in Multiobjective Function Optimization: Recommendation of GA then LS," in *2006 Genetic and Evolutionary Computation Conference (GECCO2006)*, M. K. et al., Ed., vol. 1. Seattle, Washington, USA: ACM Press. ISBN 1-59593-186-4, July 2006, pp. 667–674.

[8] A. G. Hernández-Díaz, C. A. Coello Coello, F. Pérez, R. Caballero, J. Molina, and L. V. Santana-Quintero, "Seeding the Initial Population of a Multi-Objective Evolutionary Algorithm using Gradient-Based Information," in *2008 Congress on Evolutionary Computation (CEC'2008)*. Hong Kong: IEEE Service Center, June 2008, pp. 1617–1624.

[9] A. Lara, G. Sanchez, C. A. Coello Coello, and O. Schütze, "HCS: A New Local Search Strategy for Memetic Multi-Objective Evolutionary Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 112–132, February 2010.

[10] J. Fliege and B. Fux Svaiter, "Steepest descent methods for multicriteria optimization," *Mathematical Methods of Operations Research*, vol. 51, no. 3, pp. 479–494, February 2000.

[11] S. Schäffler, R. Schultz, and K. Weinzierl, "Stochastic method for the solution of unconstrained vector optimization problems," *Journal of Optimization Theory and Applications*, vol. 114, no. 1, pp. 209–222, 2002.

[12] K. Harada, J. Sakuma, and S. Kobayashi, "Local Search for Multiobjective Function Optimization: Pareto Descent Method," in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, 2006, pp. 659–666.

[13] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance Between Genetic Search and Local Search in Memetic Algorithms for Multiobjective Permutation Flowshop Scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 204–223, April 2003.

[14] M. Dellnitz, O. Schütze, and T. Hestermeyer, "Covering Pareto Sets by Multilevel Subdivision Techniques," *Journal of optimization theory and applications*, vol. 124, no. 1, pp. 113–136, 2005.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, April 2002.

[16] H. W. Kuhn and A. W. Tucker, "Nonlinear Programming," in *Proceedings of the Second Berkeley Symposium on Mathematics Statistics and Probability*. University of California Press, 1951.

[17] P. Shukla, "Gradient Based Stochastic Mutation Operators in Evolutionary Multi-objective Optimization," in *Proceedings of the 8th international conference on Adaptive and Natural Computing Algorithms*. Warsaw, Poland: Springer. Lecture Notes in Computer Science Vol. 4431, 2007, pp. 58–66.

[18] G. Corliss, C. Faure, A. Griewank, L. Hascoët, and U. Naumann, Eds., *Automatic Differentiation of Algorithms: From Simulation to Optimization*. New York, NY, USA: Springer-Verlag New York, Inc., 2002.

[19] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York: Springer, September 2007, ISBN 978-0-387-33254-3.

[20] M. Brown and R. Smith, "Directed Multi-objective Optimization," *International Journal of Computers, Systems and Signals*, vol. 6, no. 1, pp. 3–17, 2005.

[21] X. Hu, Z. Huang, and Z. Wang, "Hybridization of the Multi-Objective Evolutionary Algorithms and the Gradient-based Algorithms," in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, vol. 2. Canberra, Australia: IEEE Press, December 2003, pp. 870–877.

[22] O. Schuetze, M. Laumanns, E. Tantar, C. A. Coello Coello, and E.-G. Talbi, "Computing Gap Free Pareto Front Approximations with Stochastic Search Algorithms," *Evolutionary Computation*, vol. 18, no. 1, pp. 65–96, Spring 2010.

[23] C. Goh, Y. Ong, K. Tan, and E. Teoh, "An investigation on evolutionary gradient search for multi-objective optimization," in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence)*, 2008, pp. 3741–3746.