A Hybridization of MOEA/D with the Nonlinear Simplex Search Algorithm

Saúl Zapotecas Martínez
CINVESTAV-IPN,
Departamento de Computación
Av. IPN 2508. San Pedro Zacatenco
México D.F. 07300, MÉXICO,
email: saul.zapotecas@gmail.com

Carlos A. Coello Coello
CINVESTAV-IPN,
Departamento de Computación
Av. IPN 2508. San Pedro Zacatenco
México D.F. 07300, MÉXICO,
email: ccoello@cs.cinvestav.mx

***Abstract*—In the last few years, the development of hybrid algorithms that combine multi-objective evolutionary algorithms (MOEAs) with mathematical programming techniques has significantly increased. Such hybrid algorithms attempt to combine the global search properties of MOEAs with the exploitative power of mathematical programming techniques. Most of these approaches normally rely on the use of gradients and, therefore, their use is limited to certain types of problems. The use of nonlinear direct search techniques—i.e., methods that do not require gradient information—has been less popular in these type of approaches. This paper focuses on the design of a hybrid algorithm between the well-known MOEA/D and one of most popular direct search methods, the Nelder and Mead method. The mathematical programming technique adopted here, acts as a local search engine, whose goal is to exploit promising regions of the search space that have been generated by MOEA/D. Our preliminary results indicate that this sort of hybridization is promising for dealing with multi-objective optimization problems having a moderately high number of decision variables.**

I. Introduction

Multi-objective evolutionary algorithms (MOEAs) have been successfully adopted for solving a wide variety of engineering and scientific problems [1]. However, one of the limitations of MOEAs is their computational cost, which turns out to be unaffordable in certain type of problems. This has motivated the development of numerous strategies for reducing the number of fitness function evaluations in a MOEA. In the last few years, several researchers have developed hybrid approaches combining MOEAs with mathematical programming techniques. Most of these multi-objective memetic algorithms require the gradient of the functions (see for example [2]), and, therefore their use is limited. In recent years, the use of direct search methods in multi-objective memetic algorithms, has attracted the attention of several researchers. Next, we briefly discuss some hybrid algorithms found in the specialized literature, which couple direct search methods to a MOEA.

In 2004, Koduru et al. [3] proposed a hybrid genetic algorithm using fuzzy dominance and nonlinear simplex search method. This approach used the simplex search as a local search engine for obtaining nondominated solutions in a genetic algorithm. The simplex was built by choosing a set of solutions from the current population. To estimate the best and the worst solution into the simplex, the fuzzy dominance relation was employed. The proposed hybrid algorithm was used to estimate the parameters of a gene regulatory network. Koduru et al. [4] hybridized nonlinear simplex search with a Multi-Objective Particle Swarm Optimizer (MOPSO). This approach adopted clustering techniques to build the simplex. The nonlinear simplex search was used as a local search engine for finding nondominated solutions in the neighborhood defined by the particle to be improved. Zapotecas and Coello [5] presented a hybridization between the well-known Nondominated Sorting Genetic Algorithm II (NSGA-II) [6] and the nonlinear simplex search method. In this approach, the search was directed by an aggregating function and the simplex was constructed using a low-discrepancy sequence within a reduced search space. The proposed memetic algorithm was tested using problems with a moderate dimensionality in decision variable space (up to 30 decision variables). Zhong et al. [7] hybridized the nonlinear simplex search and the Differential Evolution (DE) algorithm. The simplex was constructed by selecting in a random way, a set of solutions taken from the current population. The vertices of the simplex were sorted according to Pareto dominance in order to identify the worst solution into the simplex. At each iteration of the local search engine, a movement into the simplex was performed for generating new nondominated solutions. This hybrid algorithm was tested with problems having low dimensionality in decision variable space (two and three decision variables). Koch et al. [8] presented a hybrid algorithm which combined the exploratory properties of the S-Metric Selection Evolutionary Multi-objective Optimization Algorithm (SMS-EMOA) [9] with the exploitative power of the Hooke and Jeeves algorithm [10] which was used in the local search. At each iteration the Hooke and Jeeves algorithm performs an exploratory move along the coordinate axes. Afterwards, the vectors of the last exploratory moves were combined to a projected direction that can accelerate the descent of the search vector. Recently, Zapotecas and Coello [11] presented a local search mechanism for decomposition-based MOEAs (called MOEA/D+LS). This evolutionary approach hybridized the nonlinear simplex search algorithm with MOEA/D [12]. The local search engine was based on a previous extension of the nonlinear simplex search for multi-objective optimization developed by the same authors [13].

978-1-4673-5889-7/13/$31.00

In this paper, we present an improved version of MOEA/D+LS introduced previously in [11]. In analogous way, the proposed hybrid algorithm (called MOEA/D+LS-II) incorporates the Nelder and Mead method [14] as a local search engine into the well-known MOEA/D [12]. However, in order to improve the local search, some modifications have been introduced. The proposed MOEA/D+LS-II, introduces a new criteria to perform the local search procedure. Besides, the search direction and the selection mechanism (from which the local search starts) used in MOEA/D+LS are modified. As we will see later on, our preliminary results indicate that the proposed mechanisms incorporated to MOEA/D, give robustness and better performance when it is compared with respect to the original MOEA/D [12] and MOEA/D+LS [11], over a set of 21 test problem adopted.

The remainder of this paper is organized as follows. In Section II, we provide the basic definitions required for understanding the rest of the paper. Section III describes the proposed memetic algorithm, including a detailed explanation of proposed local search mechanism. Section IV presents the experimental study used for assessing the performance of our proposed memetic algorithm. In Section V, we provide a brief discussion of our results. Finally, in Section VI, we provide our conclusions and some possible paths for future research.

II. Basic Concepts

A. Multi-objective optimization

Without loss of generality and assuming minimization problems, a nonlinear multi-objective optimization problem (MOP) can be formulated as:

$$\min_{\mathbf{x}\in\Omega} \quad \mathbf{F}(\mathbf{x}) \tag{1}$$

where Ω defines the decision space and $\mathbf{F}$ is defined as the vector of objective functions: $\mathbf{F}:\Omega\rightarrow\mathbb{R}^k$, $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}),\dots,f_k(\mathbf{x}))^T$, such that $f_i:\mathbb{R}^n\rightarrow\mathbb{R}$ is a nonlinear function. In order to describe the concept of optimality in which we are interested, the following definitions are introduced [15]:

Definition 1. Let $\mathbf{x},\mathbf{y}\in\Omega$, we say that $\mathbf{x}$ *dominates* $\mathbf{y}$ (denoted by $\mathbf{x}\prec\mathbf{y}$) if and only if, $f_i(\mathbf{x})\leq f_i(\mathbf{y})$ and $f_i(\mathbf{x})<f_i(\mathbf{y})$ in at least one f_i for all $i=1,\dots,k$.
Definition 2. Let $\mathbf{x}^\star\in\Omega$, we say that $\mathbf{x}^\star$ is a *Pareto optimal* solution, if there is no other solution $\mathbf{y}\in\Omega$ such that $\mathbf{y}\prec\mathbf{x}^\star$.
Definition 3. The *Pareto optimal set* PS is defined by: $\mathcal{PS}=\{\mathbf{x}\in\Omega|\mathbf{x}\text{ is Pareto optimal solution}\}$, and the *Pareto optimal front* $\mathcal{PF}$ is defined as: $\mathcal{PF}=\{\mathbf{F}(\mathbf{x})|\mathbf{x}\in\mathcal{PS}\}$.

We thus wish to find the best possible *trade-offs* among the objectives, such that no objective can be improved without worsening another. Since the number of Pareto optimal solutions can be very large, we are also interested in obtaining a well-distributed set of solutions, since the size of our approximation (produced by a MOEA) will be normally small.

B. Decomposing Multi-Objective Optimization Problems

It is well-known that a Pareto optimal solution to a MOP, under some assumptions, is an optimal solution of a scalar optimization problem in which, the objective is an aggregation of all the objective functions f_i's. Therefore, an approximation of the Pareto optimal front can be decomposed into a number of scalar objective optimization subproblems. In the specialized literature, there are several approaches for transforming a MOP into multiple single-objective optimization subproblems [16], [15]. In the following, we briefly describe a method based on the normal boundary intersection (NBI) [17] method, which is referred to in this work.

1) Penalty Boundary Intersection Approach: The Penalty Boundary Intersection (PBI) approach proposed by Zhang and Li [12], uses a weight vector $\mathbf{w}$ and a penalty value θ for minimizing both the distance to the utopian vector d_1 and the direction error to the weight vector d_2 from the solution $\mathbf{F}(\mathbf{x})$. Therefore, the optimization problem can be stated as:

$$\text{Minimize:} \quad g(\mathbf{x}|\mathbf{w},\mathbf{z}^\star) = d_1 + \theta d_2 \tag{2}$$

where:

$$d_1 = \frac{||(\mathbf{F}(\mathbf{x})-\mathbf{z}^\star)^T\mathbf{w}||}{||\mathbf{w}||} \text{ and } d_2 = \left|\left|(\mathbf{F}(\mathbf{x})-\mathbf{z}^\star) - d_1\frac{\mathbf{w}}{||\mathbf{w}||}\right|\right|$$

such that $\mathbf{x}\in\Omega$ and $\mathbf{z}^\star=(z_1,\dots,z_k)^T$, where $z_i=\min\{f_i(\mathbf{x})|\mathbf{x}\in\Omega\}$.

Therefore, a good representation of the Pareto front can be obtained by solving a set of problems defined by a well-distributed set of weight vectors. This is the main idea behind current decomposition-based MOEAs—see for example [12], [18], [19].

C. The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D)

The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [12], transforms a MOP into several scalarization problems. Therefore, an approximation of the Pareto front is obtained by solving the N scalarization subproblems in which a MOP is decomposed.

Considering $W=\{\mathbf{w}_1,\dots,\mathbf{w}_N\}$ as the well-distributed set of weight vectors, MOEA/D finds the best solution to each subproblem defined by each weight vector using the PBI approach. The objective function of the j^{th} subproblem is then defined by $g(\mathbf{x}|\mathbf{w}_j,\mathbf{z})$, where $\mathbf{w}_j\in W$ and $\mathbf{z}=(z_1,\dots,z_k)^T$ is the artificial utopian vector whose component z_i is the minimum value found so far for the objective f_i. In MOEA/D, a neighborhood of the weight vector $\mathbf{w}_i$ is defined as a set of its closest weight vectors in W. Therefore, the neighborhood of the i^{th} subproblem consists of all the subproblems with the weight vectors from the neighborhood of $\mathbf{w}_i$ and it is denoted by $B(\mathbf{w}_i)$.

At each generation, MOEA/D finds the best solution to each subproblem throughout the evolutionary process and maintains: 1) a population of N points $P=\{\mathbf{x}_1,\dots,\mathbf{x}_N\}$, where $\mathbf{x}_i\in\Omega$ is the current solution to the i^{th} subproblem; 2) $FV^1,\dots,FV^N$, where FV^i is the F-value of $\mathbf{x}_i$,

Algorithm 1: General Framework of MOEA/D

Input:
a stopping criterion;
N: the number of the subproblems considered in MOEA/D;
W: a well-distributed set of weight vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$;
T: the number of weight vectors in the neighborhood of each weight vector.

Output:
EP: the nondominated solutions found during the search;
P: the final population found by MOEA/D.

begin
  Step 1. INITIALIZATION:
  $EP = \emptyset$;
  Generate an initial population $P = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ randomly;
  $FV^i = \mathbf{F}(\mathbf{x}_i)$;
  $B(\mathbf{w}_i) = \{\mathbf{w}_{i_1}, \ldots, \mathbf{w}_{i_T}\}$ where $\mathbf{w}_{i_1}, \ldots, \mathbf{w}_{i_T}$ are the T closest weight vectors to $\mathbf{w}_i$, for each $i = 1, \ldots, N$;
  $\mathbf{z} = (+\infty, \ldots, +\infty)^T$;
  while *stopping criterion is not satisfied* **do**
    Step 2. UPDATE: (the next population)
    for $\mathbf{x}_i \in P$ **do**
      REPRODUCTION: Randomly select two indexes k, l from $B(\mathbf{w}_i)$, and then generate a new solution $\mathbf{y}$ from $\mathbf{x}_k$ and $\mathbf{x}_l$ by using genetic operators.
      MUTATION: Apply a mutation operator on $\mathbf{y}$ to produce $\mathbf{y}'$.
      UPDATE OF $\mathbf{z}$: For each $j = 1, \ldots, k$, if $z_j < f_j(\mathbf{x})$, then set $z_j = f_j(\mathbf{y}')$.
      UPDATE OF NEIGHBORING SOLUTIONS: For each index $j \in B(\mathbf{w}_i)$, if $g(\mathbf{y}'|\mathbf{w}_j, \mathbf{z}) \leq g(\mathbf{x}_j|\mathbf{w}_j, \mathbf{z})$, then set $\mathbf{x}_j = \mathbf{y}'$ and $FV^j = \mathbf{F}(\mathbf{y}')$. UPDATE OF EP: Remove from EP all the vectors dominated by $\mathbf{F}(\mathbf{y}')$. Add $\mathbf{F}(\mathbf{y}')$ to EP if no vectors in EP dominate $\mathbf{F}(\mathbf{y}')$.
    end
  end
end

i.e., $FV^i = \mathbf{F}(\mathbf{x}_i)$ for each $i = 1, \ldots, N$; 3) an external population EP, which is used to store the nondominated solutions found during the search. Algorithm 1 presents the general framework of MOEA/D, although the interested reader can be referred to [12] for a more detailed description.

D. The Nonlinear Simplex Search

Nelder and Mead's method [14] also known as the *Nonlinear Simplex Search*, is an algorithm based on the simplex algorithm of Spendley et al. [20], which was introduced for minimizing nonlinear and multi-dimensional unconstrained functions. While Spendley et al.'s algorithm uses regular simplexes, Nelder and Mead's method generalizes the procedure to change the shape and size of the simplex. Therefore, the convergence towards a minimum value at each iteration of the nonlinear simplex search is conducted by three main movements in a geometric shape called *simplex*.

The full algorithm is defined stating three scalar parameters to control the movements performed in the simplex: **reflection** (α), **expansion** (γ) and **contraction** (β). At each iteration, the $n+1$ vertices Δ_i of the simplex represent solutions which are evaluated and sorted according to: $f(\mathbf{\Delta}_1) \leq f(\mathbf{\Delta}_2) \leq \cdots \leq f(\mathbf{\Delta}_{n+1})$. In this way, the movements performed in the simplex by the nonlinear simplex search method are defined as:

1) *Reflection*: $\mathbf{x}_r = (1+\alpha)\mathbf{x}_c - \alpha\mathbf{\Delta}_{n+1}$.
2) *Expansion*: $\mathbf{x}_e = (1+\alpha\gamma)\mathbf{x}_c - \alpha\gamma\mathbf{\Delta}_{n+1}$.
3) *Contraction*:
   a) *Outside*: $\mathbf{x}_{co} = (1+\alpha\beta)x_c - \alpha\beta\mathbf{\Delta}_{n+1}$.
   b) *Inside*: $\mathbf{x}_{ci} = (1-\beta)\mathbf{x}_c + \beta\mathbf{\Delta}_{n+1}$.

where $\mathbf{x}_c = \frac{1}{n}\sum_{i=1}^{n} \Delta_i$ is the centroid of the n best points (all vertices except for $\mathbf{\Delta}_{n+1}$), $\mathbf{\Delta}_1$ and $\mathbf{\Delta}_{n+1}$ are the best and the worst solutions identified within the simplex, respectively. At each iteration, the initial simplex is modified by one of the above movements, according to the following rules:

1. If $f(\mathbf{\Delta}_1) \leq f(\mathbf{x}_r) \leq f(\mathbf{\Delta}_n)$, then $\mathbf{\Delta}_{n+1} = \mathbf{x}_r$.
2. If $f(\mathbf{x}_e) < f(\mathbf{x}_r) < f(\mathbf{\Delta}_1)$, then $\mathbf{\Delta}_{n+1} = x_e$, otherwise $\mathbf{\Delta}_{n+1} = \mathbf{x}_r$.
3. If $f(\mathbf{\Delta}_n) \leq f(\mathbf{x}_r) < f(\mathbf{\Delta}_{n+1})$ and $f(\mathbf{x}_{co}) \leq f(\mathbf{x}_r)$, then $\mathbf{\Delta}_{n+1} = \mathbf{x}_{co}$.
4. If $f(\mathbf{x}_r) \geq f(\mathbf{\Delta}_{n+1})$ and $f(\mathbf{x}_{ci}) < f(\mathbf{\Delta}_{n+1})$, then $\mathbf{\Delta}_{n+1} = \mathbf{x}_{ci}$.

III. THE PROPOSED APPROACH

A. General Framework

Our proposed multi-objective memetic algorithm adopts MOEA/D [12] as its baseline algorithm. The proposed local search mechanism is based on Nelder and Mead's method [14]. The memetic algorithm (called here, MOEA/D+LS-II) explores the global search space using MOEA/D, while the local search engine exploits the promising regions provided by the MOEA. For a better understanding of the proposed approach, Algorithm 2 presents the general framework of the proposed MOEA/D+LS-II. **Step 3** refers to the complete local search mechanism which is performed after each iteration of MOEA/D. In the following sections, we describe in detail the components of the local search mechanism.

B. Local Search Mechanism

MOEA/D+LS-II exploits the promising neighborhood of the solutions found by the MOEA at each generation. As it was mentioned before, MOEA/D+LS-II uses Nelder and Mead's method as a local search engine for continuous search spaces, in order to improve the solutions provided by MOEA/D. In

Algorithm 2: General Framework of MOEA/D+LS-II

Input:
a stopping criterion;
N: the number of the subproblems considered in MOEA/D+LS-II;
W: a well-distributed set of weight vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$;
T: the number of weight vectors in the neighborhood of each weight vector;
S_t: the similarity threshold for the local search;
E_{ls}: the maximum number of evaluations for the local search.

Output:
EP: the nondominated solutions found during the search;
P: the final population found by MOEA/D+LS-II.

begin
  Step 1. INITIALIZATION:
  Generate an initial population $P = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ randomly;
  $FV^i = \mathbf{F}(\mathbf{x}_i)$;
  $B(\mathbf{w}_i) = \{\mathbf{w}_{i_1}, \ldots, \mathbf{w}_{i_T}\}$ where $\mathbf{w}_{i_1}, \ldots, \mathbf{w}_{i_T}$ are the T closest weight vectors to $\mathbf{w}_i$, for each $i = 1, \ldots, N$;
  $\mathbf{z} = (+\infty, \ldots, +\infty)^T$;
  Step 2. THE MEMETIC ALGORITHM:
  while *stopping criterion is not satisfied* **do**
    Perform **Step 2 of the MOEA/D** algorithm for obtaining P (the next population).
    Step 3. THE LOCAL SEARCH MECHANISM:
    for $j = 1, \ldots, k+1$ **do**
      Step 3.1. DEFINING THE SEARCH DIRECTION:.
      if $j < k$ **then**
        // Search towards the extremes of the Pareto front
        $\mathbf{w_s} = \mathbf{e}^j$, where $\mathbf{e}^j$ is the j^{th} canonical basis in $\mathbb{R}^k$ and k is the number of objective functions.
      else
        // Search towards the maximum bulge of the Pareto front
        $\mathbf{w_s} = (1/k, \ldots, 1/k)$
      end
      Step 3.2. SELECTING INITIAL SOLUTION: Select the initial solution for the local search according to Section III-B2.
      Step 3.3. LOCAL SEARCH: Apply nonlinear simplex search according to the Algorithm 3.
    end
  end
end

Algorithm 3: Use of Local Search

Input:
a stopping criterion;
S_t: the similarity threshold for the local search;
E_{ls}: the maximum number of evaluations for the local search.

Output:
P: the updated population P.

begin
  Step 1. CHECKING SIMILARITY: Obtain the similarity (S_{ls}) between $\mathbf{p}_{ini}$ and the previous initial solution ($\mathbf{p}'_{ini}$) for the local search—see Section III-B3;
  if *there are enough resources* **and** $S_t < S_{ls}$ **then**
    Step 2. BUILDING THE SIMPLEX: Build the initial simplex for the nonlinear simplex search—see Section III-B4;
    Step 3. DEFORMING THE SIMPLEX: Perform any movement (reflection, contraction or expansion) for obtaining $\mathbf{p}_{new}$ according to Nelder and Mead's method—see Section III-B5;
    Step 4. UPDATING THE POPULATION: Update the population P using the new solution $\mathbf{p}_{new}$ according to the rules presented in Section III-B6.
    Step 5. STOPPING CRITERION: If the stopping criterion is satisfied then stop the local search. Otherwise go to **Step 3**—see Section III-B7.
  end
end

contrast to MOEA/D+LS [11], the local search mechanism of MOEA/D+LS-II approximates solutions to the extremes and the maximun bulge (sometimes called knee) of the Pareto front. The nonlinear simplex search is employed for minimizing a subproblem defined by a weight vector using the PBI approach. In the following, we present in detail the components of our local search engine outlined in Algorithms 2 and 3.

1) Defining the Search Direction: In contrast to the method proposed in [11], the local search mechanism proposed here, approximates solutions to the Pareto front in two different stages.

1) Initially, the search is directed to the extremes of the Pareto front. Therefore, the weight vectors that define the subproblems that approximate solutions (when they are solved) to the extremes are defined by the canonical basis in $\mathbb{R}^k$—i.e., the search direction that approximates solutions to the j^{th} extreme of the Pareto front is defined by the weight vector: [1]
$$\mathbf{w}_s = \mathbf{e}^j$$
where $\mathbf{e}^j$ is the j^{th} canonical vector in $\mathbb{R}^k$.
2) Once the solutions lying at the extremes of the Pareto front have been approximated, the local search is focused on minimizing the subproblem that approximates the solutions lying on the knee of the Pareto front. Therefore, the search direction is now defined by the weight vector:
$$\mathbf{w}_s = (1/k, \ldots, 1/k)^T$$
where k is the number of objective functions.

Considering the use of the PBI approach, the penalty value θ is set as $\theta = 5$ for approximating solutions to the extremes,

[1]Assuming the use of the PBI approach.

whereas for the knee, a value $\theta = 10$ is employed. Note that the search on the knee is relaxed defining a bigger θ value than the one stated for the extremes.

2) Selection Mechanism: Let P be the set of solutions found by MOEA/D at any generation. Let $\mathbf{w}_s$ be the weight vector that defines the search direction for the nonlinear simplex search. The solution $\mathbf{p}_{ini}$ which starts the search is defined by:

$$\mathbf{p}_{ini} = \mathbf{x} \in P, \text{ such that minimizes: } g(\mathbf{x}|\mathbf{w}_s, \mathbf{z}^\star)$$

Solution $\mathbf{p}_{ini}$ represents not only the initial search point, but also the simplex head from which the simplex will be built.

3) Checking Similarity: The nonlinear simplex search explores the neighborhood of the solution $\mathbf{p}_{ini} \in P$. Since the simplex search is applied after each iteration of the MOEA, most of the time, the initial solution $\mathbf{p}_{ini}$ does not change its position from one generation to another. For this reason, the proposed local search mechanism stores a record ($\mathbf{p}'_{ini}$) of the last position from which the nonlinear simplex search starts. At the beginning of the execution of MOEA/D+LS-II, the initial position record is set as empty, that is: $\mathbf{p}'_{ini} = \emptyset$. Once the simplex search is performed, the initial solution is stored in the historical record, i.e., $\mathbf{p}'_{ini} = \mathbf{p}_{ini}$. In this way, for the next call of the local search, a previous comparison of similarity is performed. That is, the local search will be performed, if and only if, $||\mathbf{p}_{ini} - \mathbf{p}'_{ini}|| > S_{ls}$, where S_{ls} represents the similarity threshold. Since in the first iteration of the simplex search, there is no previous record of the initial solution, the simplex search is automatically performed. Both the updating of the historical record and the similarity operator are performed for each initial solution $\mathbf{p}_{ini}$ which minimizes the subproblem defined by $\mathbf{w}_s$. In our study, we adopted a similarity threshold $S_{ls} = 0.001$. This strategy differs from those presented in [11], where local search is applied when the population has less than 50% of nondominated solutions.

4) Building the Simplex: Let $\mathbf{w}_{ini}$ be the weight vector that defines the subproblem for which the initial search point $\mathbf{p}_{ini}$ is minimum. Let $S(\mathbf{w}_{ini})$ be the neighborhood of the n closest weight vectors to $\mathbf{w}_{ini}$ (where n is the number of decision variables of the MOP). Then, the simplex defined as:

$$\mathbf{\Delta} = \{\mathbf{p}_{ini}, \mathbf{p}_1, \ldots, \mathbf{p}_n\}$$

is built in two different ways, depending on the direction on which the simplex search is focused.

i. *For the extremes of the Pareto front:* The remaining n solutions $\mathbf{p}_i \in \Omega$ $(i = 1, \ldots, n)$ are generated by using a low-discrepancy sequence. In this work, we adopted the Hammersley sequence [21] to generate a well-distributed sampling of solutions in a determined search space. As in [11], we use a strategy based on the genetic analysis of a sample from the current population for reducing the search space. Therefore, we compute the average ($\mathbf{m}$) and standard deviation (σ) of the chromosomes (solutions) that minimize each subproblem defined by the weight vectors in $S(\mathbf{w}_{ini})$. In this way, the new bounds are defined by:

$$\begin{aligned} \mathbf{L}_{bound} &= \mathbf{m} - \sigma \\ \mathbf{U}_{bound} &= \mathbf{m} + \sigma \end{aligned}$$

where $\mathbf{L}_{bound}$ and $\mathbf{U}_{bound}$ are the vectors which define the lower and upper bounds of the new search space, respectively. Once the search space has been reduced, the n remaining solutions are generated by means of the Hammersley sequence using as bounds $\mathbf{L}_{bound}$ and $\mathbf{U}_{bound}$.

ii. *For the knee of the Pareto front:* The remaining n solutions $\mathbf{p}_i \in P$ $(i = 1, \ldots, n)$ are chosen, such that, $\mathbf{p}_i$ minimizes each subproblem defined by each weight vector in $S(\mathbf{w}_{ini})$. This is the same strategy employed in MONSS [13] for constructing the simplex.

Note however that, since the dimensionality of the simplex depends of the number of decision variables of the MOP, the population size of the MOEA needs to be larger than the number of decision variables.

5) Deforming the Simplex: Let $\mathbf{w}_s$ be the weight vector that defines the search direction for the local search. Let $\mathbf{\Delta}$ be the simplex defined by the above description. The nonlinear simplex search will be focused on minimizing the subproblem defined by the weight vector $\mathbf{w}_s$. At each iteration of the nonlinear simplex search, the $n + 1$ vertices of the simplex $\mathbf{\Delta}$ are sorted according to their value for the subproblem that it tries to minimize (the best value is the first element). In this way, a movement into the simplex is performed for generating the new solution $\mathbf{p}_{new}$. The movements are calculated according to the equations provided by Nelder and Mead—see Section II-D. Each movement is controlled by three scalar parameters: reflection (α), expansion (β) and contraction (γ).

The simplex search was conceived for unbounded problems. When dealing with bounded variables, the created solutions can be located outside the allowable bounds after some movements of the simplex search. In order to deal with this, we bias the new solution if any component of $\mathbf{p}_{new}$ lies outside the bounds according to:

$$\mathbf{p}_{new}^{(j)} = \begin{cases} \mathbf{L}_{bound}^{(j)} & \text{, if } \mathbf{p}_{new}^{(j)} < \mathbf{L}_{bound}^{(j)} \\ \mathbf{U}_{bound}^{(j)} & \text{, if } \mathbf{p}_{new}^{(j)} > \mathbf{U}_{bound}^{(j)} \\ \mathbf{p}_{new}^{(j)} & \text{, otherwise.} \end{cases} \tag{3}$$

where $\mathbf{L}_{bound}^{(j)}$ and $\mathbf{U}_{bound}^{(j)}$ are the lower and upper bounds of the j^{th} parameter of $\mathbf{p}_{new}$, respectively.

6) Updating the Population: The information provided by the local search mechanism is introduced into the population of MOEA/D. Since we are dealing with MOPs, the new solution generated by any movement of the nonlinear simplex search could be better than more than one solution in the current population. Thus, we adopt the following mechanism in which more than one solution from the population could be replaced.

Let P be the current population reported by the MOEA. Let $\mathbf{p}_{new}$ be the solution generated by any movement of the simplex search. Let $B(\mathbf{w}_s)$ and $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ be the neighborhood of the T closest weight vectors to $\mathbf{w}_s$, and the well-distributed set of all weight vectors, respectively. We define

$$Q = \begin{cases} B(\mathbf{w}_s) & \text{, if } r < \delta \\ W & \text{otherwise} \end{cases}$$

where r is a random number having uniform distribution. In this work, we use $\delta = 0.5$.

The current population P is updated by replacing at most R_{ls} solutions from P such that, $g(\mathbf{p}_{new}|\mathbf{w}_i, z) < g(\mathbf{x}_i|\mathbf{w}_i, z)$, where $\mathbf{w}_i \in Q$ and $\mathbf{x}_i \in P$, such that $\mathbf{x}_i$ minimizes the subproblem defined by $\mathbf{w}_i$.

In this way, the loss of diversity is avoided by replacing a maximum number of solutions from P, instead of all the solutions that minimize the subproblems defined by the complete neighborhood Q. In our study, we set $R_{ls} = 15$ as the maximum number of solution to replace.

7) Stopping Criterion: The local search mechanism encompasses the search of solutions towards both the extremes and the knee of the Pareto front. This mechanism is limited to a maximum number of fitness function evaluations defined by E_{ls}. In this way, the proposed local search has the following stopping criteria:

1) If the nonlinear simplex search overcomes the maximum number of evaluations (E_{ls}), the simplex search is stopped and the evolutionary process of MOEA/D continues by going to **Step 2** of Algorithm 1.
2) The search could be inefficient if the simplex has been deformed so that it has collapsed into a region in which there are no local minima. According to Lagarias et al. [22] the simplex search finds a better solution in at most $n + 1$ iterations (at least in convex functions with low dimensionality). Therefore, if the simplex search does not find a better value for the subproblem defined by $\mathbf{w}_s$ in $n+1$ iterations, we stop the search and continue with the next direction defined by going to **Step 3.1** of Algorithm 2. Otherwise, we perform other movement into the simplex by going to **Step 3** of Algorithm 3.

IV. Experimental Results

A. Test Problems and Performance Assessment

In order to assess the performance of our proposed memetic algorithm, we compare its results with respect to those obtained by the original MOEA/D [12] and the state-of-the-art MOEA/D+LS [11]. We adopted 21 test problems whose Pareto fronts have different characteristics including convexity, concavity, disconnections and multi-modality. In the following, we describe the test suites that we have adopted.

- **Zitzler-Deb-Thiele** (ZDT) test suite [23]. The four bio-objective MOPs (except for ZDT5, which is a discrete problem) were adopted. We used 30 decision variables for ZDT1 to ZTD3, while ZDT4 and ZDT6 were tested using 10 decision variables.
- **Deb-Thiele-Laumanns-Zitzler** (DTLZ) test suite [24]. The seven unconstrained MOPs were adopted. DTLZ1 was tested using 7 decision variables. For DTLZ2 to DTLZ6, we employed 12 decision variables, while DTLZ7 was tested using 22 decision variables. For all problems we tested the algorithms using three objective functions for each MOP.
- **Working-Fish-Group** (WFG) test suite [25]. The nine MOPs from this test suite were adopted. We used $k = 4$ for the position related parameters and $l = 20$ for the distance related parameters—i.e. 24 decision variables (as it was suggested in [25])—adopting three objective functions for each MOP.

To assess the performance of our proposed memetic algorithm and the other two state-of-the-art MOEAs (i.e., the original MOEA/D and MOEA/D+LS) on the test problems adopted, the **Hypervolume** (I_H) indicator was employed [26]. This performance measure is Pareto compliant [27], and quantifies both approximation and maximum spread of nondominated solutions along the Pareto front. The interested reader is referred to [26] for a more detailed description of this metric.

B. Parameters Settings

We compared the results obtained by our proposed MOEA/D+LS-II with respect to those obtained by MOEA/D and MOEA/D+LS (using the PBI approach). For a fair comparison, the set of weight vectors was the same for all the algorithms, and they were generated in the same way as in [12]. Although, other methods can be also used, see e.g. [28], [29]. For each MOP, 30 independent runs were performed with each algorithm. The parameters for the algorithms are summarized in Table I, where N represents the number of initial solutions (100 for bi-objective problems and 300 for three-objective problems). N_{it} represents the maximum number of iterations, which was set to 100 for all test problems. Therefore, both algorithms performed 10,000 (for the bi-objective problems) and 30,000 (for the three-objective problems) fitness function evaluations for each problem. The parameters T_n, η_c, η_m, P_c and P_m represent the neighborhood size, crossover index (for Simulated Binary Crossover (SBX)), mutation index (for Polynomial-Based Mutation (PBM)), crossover rate and mutation rate, respectively. For MOEA/D+LS and MOEA/D+LS-II, α, β and γ represent the control parameters for the reflection, expansion and contraction movements of the nonlinear simplex search, respectively. R_{ls} and E_{ls} represent the number of solutions to be replaced and the maximum number of fitness function evaluations employed by the local search, respectively. A_r and S_{ls}, represent the action range and the similarity threshold employed by the local search for MOEA/D+LS and MOEA/D+LS-II, respectively. Finally, the parameter θ, represents the penalty value used in the PBI approach for the three approaches compared herein.

For each MOP, the algorithms were evaluated using the Hypervolume (I_H) indicator. The results obtained are summarized in Table II. These tables display both the $average$ and the standard deviation (σ) of the I_H indicator for each MOP. The reference vectors used for computing the I_H performance measure are shown in Table II. These vectors are established close to the individual minima for each MOP, i.e., close to the extremes of the Pareto optimal front. With

TABLE I
PARAMETERS FOR MOEA/D, MOEA/D+LS AND MOEA/D+LS-II

Parameter	MOEA/D	MOEA/D+LS	MOEA/D+LS-II
N	100/300	100/300	100/300
N_{it}	100	100	100
T_n	20	20	20
η_c	20	20	20
η_m	20	20	20
P_c	1	1	1
P_m	$1/n$	$1/n$	$1/n$
α	–	1	1
β	–	2	2
γ	–	1/2	1/2
R_{ls}	–	15	15
E_{ls}	–	300	300
A_r	–	5	–
S_{ls}	–	–	0.001
θ	5	5	5

that, a good measure of approximation and spread is reported when the algorithms converge along the Pareto front. For an easier interpretation, the best results are presented in **boldface** for each test problem adopted.

V. Discussion of Results

As indicated before, the results obtained by our proposed memetic algorithm (i.e., the MOEA/D+LS-II) were compared against those produced by the original MOEA/D and MOEA/D+LS. According to the results presented in Table II, MOEA/D+LS-II had a better performance than MOEA/D and MOEA/D+LS in most of the MOPs adopted. This table provides a quantitative assessment of the performance of MOEA/D+LS-II in terms of the I_H indicator. That means that the solutions obtained by MOEA/D+LS-II achieved a better approximation of the Pareto optimal front than those solutions obtained by both MOEA/D and MOEA/D+LS-II when a low number of fitness function evaluations was used.

Note however, that for ZDT4, DTLZ1, DTLZ3 and WFG2, the I_H indicator showed that the local search mechanisms employed by both MOEA/D+LS and MOEA/D+LS-II did not improve the performance of the original MOEA/D. The poor performance of these hybrid MOEAs for ZDT4, DTLZ1 and DTLZ3 is attributed to the high multi-frontality that these problems have—for a detailed description of these problems see [23], [24]. Analogously, the multi-modality of WFG2 (presented in the last function of the MOP) has an influence on the performance of the hybrid MOEAs—for a detailed description of the WFG test suite see [25].

The effectiveness of MOEA/D+LS with respect to the original MOEA/D in the ZDT and DTLZ test suites has been shown in [11]. The proposed MOEA/D+LS-II presented here, was compared with respect to MOEA/D+LS not only in the ZDT and DTLZ test suites, but also adopting the WFG test suite. In Table II, it is possible to see that the proposed MOEA/D+LS-II outperformed MOEA/D+LS in most of the test problems adopted. Our results indicate that MOEA/D+LS-II obtained better results in the ZDT and DTLZ test problems. However, for ZDT2 and DTLZ4, our proposed approach was outperformed by MOEA/D+LS, but not in a significant manner. Thus, we argue that both MOEA/D+LS and MOEA/D+LS-II are competitive in the ZDT and DTLZ test suites. Regarding the WFG test suite, MOEA/D+LS-II showed its robustness outperforming both to MOEA/D+LS, and the original MOEA/D in most problems, which are considered more difficult to solve [25].

VI. Conclusions and Future Work

We have proposed a hybridization of MOEA/D with a nonlinear simplex search method, in which the mathematical programming method works as a local search engine. The local search mechanism approximates solutions to the extremes and the maximum bulge of the Pareto front adopting a decomposition approach. Therefore, its use could be easily coupled within other decomposition-based MOEAs, such as those reported in [18], [19]. Our proposed multi-objective memetic algorithm was found to be competitive with respect to the original MOEA/D and the MOEA/D+LS over a set of test functions taken from the specialized literature, when performing 10,000 and 30,000 fitness function evaluations, for problems having two and three objectives, respectively. We consider that the proposed strategy employed to hybridize Nelder and Mead's method with MOEA/D was appropriate for dealing with the MOPs adopted here.

As part of our future work, we intend to focus on designing other mechanism that helps us decide whether the local search engine will be triggered or not. We also plan to explore different strategies for constructing the simplex. We believe that the use of an appropriate simplex and a good hybridization strategy could be a powerful combination for solving complex and computationally expensive MOPs—as for example those presented in [12]. Finally, we also aim to extend our hybrid approach to constrained MOPs using any variants of the nonlinear simplex search algorithm for dealing with such problems.

Acknowledgments

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

References

[1] 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.

[2] P. K. Shukla, "On Gradient Based Local Search Methods in Unconstrained Evolutionary Multi-objective Optimization," in *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.

[3] P. Koduru, S. Das, S. Welch, and J. L. Roe, "Fuzzy Dominance Based Multi-objective GA-Simplex Hybrid Algorithms Applied to Gene Network Models," in *GECCO'2004*, K. D. et al., Ed. Seattle, Washington, USA: Springer-Verlag, Lecture Notes in Computer Science Vol. 3102, June 2004, pp. 356–367.

[4] P. Koduru, S. Das, and S. M. Welch, "Multi-Objective Hybrid PSO Using ϵ-Fuzzy Dominance," in *GECCO'2007*, D. Thierens, Ed., vol. 1. London, UK: ACM Press, July 2007, pp. 853–860.

TABLE II
RESULTS OF I_H INDICATOR FOR MOEA/D, MOEA/D+LS AND MOEA/D+LS-II

MOP	MOEA/D *average* (σ)	MOEA/D+LS *average* (σ)	MOEA/D+LS-II *average* (σ)	reference vector **r**
ZDT1	0.751315 (0.033339)	0.819246 (0.038088)	**0.842309** (0.009087)	$(1.1, 1.1)^T$
ZDT2	0.210410 (0.080132)	**0.384962** (0.151212)	0.363225 (0.133365)	$(1.1, 1.1)^T$
ZDT3	0.990212 (0.089499)	0.995692 (0.158499)	**1.055714** (0.230182)	$(1.1, 1.1)^T$
ZDT4	**0.600217** (0.138989)	0.169257 (0.212639)	0.185765 (0.156602)	$(1.1, 1.1)^T$
ZDT6	0.425904 (0.010630)	0.462559 (0.050484)	**0.462714** (0.022012)	$(1.1, 1.1)^T$
DTLZ1	**0.317249** (0.000957)	0.316904 (0.001091)	0.317083 (0.001075)	$(0.7, 0.7, 0.7)^T$
DTLZ2	0.768696 (0.000644)	0.768621 (0.000466)	**0.768727** (0.000594)	$(1.1, 1.1, 1.1)^T$
DTLZ3	**0.383622** (0.245603)	0.221197 (0.282045)	0.128942 (0.219193)	$(1.1, 1.1, 1.1)^T$
DTLZ4	0.768935 (0.000645)	**0.768966** (0.000664)	0.768122 (0.000574)	$(1.1, 1.1, 1.1)^T$
DTLZ5	0.426115 (0.000675)	0.426307 (0.000167)	**0.426492** (0.000114)	$(1.1, 1.1, 1.1)^T$
DTLZ6	0.000228 (0.001226)	0.426345 (0.000714)	**0.426416** (0.000254)	$(1.1, 1.1, 1.1)^T$
DTLZ7	1.916040 (0.016969)	1.922224 (0.012057)	**1.929710** (0.162598)	$(1.1, 1.1, 6.1)^T$
WFG1	14.964720 (1.030077)	15.921475 (0.955856)	**16.510348** (0.202859)	$(3, 4, 4)^T$
WFG2	**8.996212** (0.964342)	8.973534 (0.857198)	8.882838 (0.822917)	$(2, 2, 4)^T$
WFG3	39.740488 (1.120458)	39.594021 (0.987888)	**40.721010** (0.745928)	$(4, 3, 6)^T$
WFG4	69.160679 (0.877216)	**69.193123** (1.001366)	68.763272 (0.993944)	$(3, 5, 7)^T$
WFG5	65.818947 (0.828483)	**66.050850** (0.727933)	65.825280 (0.525636)	$(3, 5, 7)^T$
WFG6	65.712844 (1.167871)	64.694658 (2.015885)	**66.323221** (0.364806)	$(3, 5, 7)^T$
WFG7	66.490864 (1.388620)	66.844937 (1.478663)	**67.179656** (0.141568)	$(3, 5, 7)^T$
WFG8	62.742809 (1.249541)	62.880565 (1.148814)	**62.988349** (0.229227)	$(3, 5, 7)^T$
WFG9	63.019018 (1.486697)	62.835454 (2.171200)	**64.601092** (0.437234)	$(3, 5, 7)^T$

[5] S. Zapotecas Martínez and C. A. Coello Coello, “A Proposal to Hybridize Multi-Objective Evolutionary Algorithms with Non-Gradient Mathematical Programming Techniques,” in *PPSN X*, vol. 5199. Springer. Lecture Notes in Computer Science, September 2008, pp. 837–846.

[6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II.” *IEEE Transactions Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[7] X. Zhong, W. Fan, J. Lin, and Z. Zhao, “Hybrid non-dominated sorting differential evolutionary algorithm with nelder-mead,” in *Intelligent Systems (GCIS), 2010 Second WRI Global Congress on*, vol. 1, December 2010, pp. 306 –311.

[8] P. Koch, O. Kramer, G. Rudolph, and N. Beume, “On the hybridization of sms-emoa and local search for continuous multiobjective optimization,” in *GECCO’2009*. New York, NY, USA: ACM, 2009, pp. 603–610.

[9] N. Beume, B. Naujoks, and M. Emmerich, “Sms-emoa: Multiobjective selection based on dominated hypervolume,” *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[10] R. Hooke and T. A. Jeeves, ““Direct Search” Solution of Numerical and Statistical Problems,” *J. ACM*, vol. 8, no. 2, pp. 212–229, 1961.

[11] S. Zapotecas Martínez and C. A. Coello Coello, “A Direct Local Search Mechanism for Decomposition-based Multi-Objective Evolutionary Algorithms,” in *CEC’2012*. Brisbane, Australia: IEEE Press, June 2012, pp. 3431–3438.

[12] Q. Zhang and H. Li, “MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, December 2007.

[13] S. Zapotecas Martínez and C. A. Coello Coello, “MONSS: A Multi-Objective Nonlinear Simplex Search Algorithm,” Evolutionary Computation Group at CINVESTAV, Departamento de Computacin, CINVESTAV-IPN, México, Tech. Rep. EVOCINV-01-2013, February 2013. [Online]. Available: http://delta.cs.cinvestav.mx/ ccoello/techreports/evocinv-01-2013.pdf.gz

[14] J. A. Nelder and R. Mead, “A Simplex Method for Function Minimization,” *The Computer Journal*, vol. 7, pp. 308–313, 1965.

[15] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, Massachuisetts: Kluwer Academic Publishers, 1999.

[16] P. Vincke, *Multicriteria Decision-Aid*. John Wiley & Sons, New York, 1992.

[17] I. Das and J. E. Dennis, “Normal-boundary intersection: a new method for generating Pareto optimal points in multicriteria optimization problems,” *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[18] W. Peng and Q. Zhang, “A decomposition-based multi-objective particle swarm optimization algorithm for continuous optimization problems,” in *IEEE International Conference on Granular Computing, 2008. GrC 2008*, 2008, pp. 534 –537.

[19] S. Zapotecas Martínez and C. A. Coello Coello, “A Multi-objective Particle Swarm Optimizer Based on Decomposition,” in *GECCO’2011*. Dublin, Ireland: ACM Press, July 2011, pp. 69–76.

[20] W. Spendley, G. R. Hext, and F. R. Himsworth, “Sequential Application of Simplex Designs in Optimization and Evolutionary Operation,” *Technometrics*, vol. 4, no. 4, pp. 441–461, November 1962.

[21] J. M. Hammersley, “Monte-Carlo methods for solving multivariable problems,” *Annals of the New York Academy of Science*, vol. 86, pp. 844–874, 1960.

[22] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, “Convergence properties of the Nelder–Mead simplex method in low dimensions,” *SIAM Journal of Optimization*, vol. 9, pp. 112–147, 1998.

[23] E. Zitzler, K. Deb, and L. Thiele, “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results,” *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, Summer 2000.

[24] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable Test Problems for Evolutionary Multiobjective Optimization,” in *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, A. Abraham, L. Jain, and R. Goldberg, Eds. USA: Springer, 2005, pp. 105–145.

[25] S. Huband, P. Hingston, L. Barone, and L. While, “A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit,” *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, October 2006.

[26] E. Zitzler and L. Thiele, “Multiobjective Optimization Using Evolutionary Algorithms—A Comparative Study,” in *PPSN V*, A. E. Eiben, Ed. Amsterdam: Springer-Verlag, September 1998, pp. 292–301.

[27] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, “Performance Assessment of Multiobjective Optimizers: An Analysis and Review,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, April 2003.

[28] I. Das, “Nonlinear multicriteria optimization and robust optimality,” Ph.D. dissertation, Rice University, Houston, Texas, 1997.

[29] Y. yan Tana, Y. chang Jiaoa, H. Lib, and X. kuan Wanga, “MOEA/D + uniform design: A new version of MOEA/D for Optimization Problems with Many Objectives,” *Computers & Operations Research*, 2012.