On the Low-Discrepancy Sequences and their Use in MOEA/D for High-Dimensional Objective Spaces

Saúl Zapotecas-Martínez*, Hernán Aguirre†, Kiyoshi Tanaka‡ and Carlos A. Coello Coello§
*†‡ Faculty of Engineering, Shinshu University
4-17-1 Wakasato, Nagano 380-8553, Japan.
§CINVESTAV-IPN (Evolutionary Computation Group),
Departamento de Computación, México D.F. 07360, MÉXICO
Email: {zapotecas,ahernan,ktanaka}@shinshu-u.ac.jp, ccoello@cs.cinvestav.mx

***Abstract*—In spite of the success of the multi-objective evolutionary algorithm based on decomposition (MOEA/D), the generation of weights for problems having many objectives, continues to be an open research problem. In this paper, we introduce a new methodology based on low-discrepancy sequences to generate the weights vectors employed by MOEA/D. We analyze and compare the proposed methodology using different low-discrepancy sequences and its impact in the search process of MOEA/D. The proposed approach is evaluated in problems having many objective functions (up to 15 objectives). We show the flexibility and ease of use of this type of sequences when adopting them to generate the weights of MOEA/D.**

I. Introduction

Multi-objective evolutionary algorithms (MOEAs) have been successfully adopted for solving a wide variety of engineering and scientific problems [1]. Since their origins, MOEAs have had two main goals: 1) to find as many (different) Pareto optimal solutions as possible; and 2) to maintain a proper representation of the Pareto front (i.e., a well-distributed set of solutions along the Pareto front). During several years, MOEAs adopted selection mechanisms based on Pareto optimality [2] combined with a density estimator (responsible of distributing solutions along the Pareto front) based on techniques such as: clustering [3], crowding distance [4], adaptive grids [5], niching and fitness sharing [6], among many others. More recently, a multi-objective evolutionary algorithm based on decomposition (MOEA/D) was introduced [7]. MOEA/D explicitly decomposes a multi-objective optimization problem (MOP) into a finite number of single-objective optimization subproblems. Each subproblem is defined by a scalarization function and the emphasis on each objective is determined by a weights vector. As it is well known, a Pareto optimal solution of a MOP (under certain conditions) is an optimal solution of a scalar optimization problem [8]. Therefore, different elements of the Pareto set can be reached by optimizing different scalarization functions defined by different weights vectors. The uniformity of the weights vectors, usually reflects the uniformity of the solutions produced by MOEAs based on decomposition. However, when the number of objectives increase, the uniformity of weights vectors becomes a problem which deserves to be studied and it is precisely the focus of the study reported herein.

In the specialized literature, there are several methods that have been used to generate a set of weights vectors. Among these techniques, the simplex-lattice design [9] has become one of the most adopted methodologies in multi-objective optimization algorithms—see for example [7]. Simplex-lattice design is, perhaps, the best strategy employed to generate evenly distributed weights vectors. However, it has a disadvantage: the number of weights vectors increases in a binomial way with respect to the dimension of the weights (k) and the number of different coefficients (H) used in the design. The number of weights vectors generated by this strategy is given by $\binom{H+k-1}{k}$. In this way, when $k = 3$ and $H = 10$, the simplex-lattice design will generate 66 weights vectors, while for $k = 10$ and $H = 10$, a total of $92,378$ weight vectors will be generated. In fact, the use of a high number of weights vectors increases the computational complexity of multi-objective optimization algorithms and in some applications, they become impractical. Although other similar methods have been proposed—for example the simplex-centroid design [10] and the axial design [11] method—they have the same problem, the higher the dimensionality of the weights, the higher the number of weights vectors.

In this paper, we analyze some low-discrepancy sequence and their use in the construction of weights vectors. The proposed approach is compared with respect to one strategy which uses the simplex-lattice design to generate evenly distributed weights in two different layers. We illustrate the flexibility, ease of use and low complexity of our proposed methodology, when used to generate a set of weights vectors in high-dimensional spaces. As will be seen later on, the distribution of weights vectors produced by our proposed approach remains appropriate even when the dimensionality of the weights increases. In addition, we analyze the impact on the search process of MOEA/D when adopting the weights vectors generated by our proposed methodology.

The remainder of this paper is organized as follows. In Section II we introduce the basic terminology used in this paper. Section III presents basic concepts related to the discrepancy theory and the low-discrepancy sequences. In Section IV, we describe in detail the proposed methodology to generate the set

The last author acknowledges support from CONACyT project no. 221551.

of weights vectors. Section V shows the experimental analysis of our proposed approach. Finally, in Section VI, we provide our final remarks.

II. General Background

A. Multi-objective Optimization

Assuming minimization, a continuous multi-objective optimization problem (MOP), can be stated as:

$$\underset{\mathbf{x}\in\Omega}{\text{minimize}} \quad \mathbf{F}(\mathbf{x}) \tag{1}$$

where $\Omega \subset \mathbb{R}^n$ defines the decision space and $\mathbf{F}$ is defined as the vector of the objective functions: $\mathbf{F} : \Omega \to \mathbb{R}^k$, $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^\intercal$ where each $f_j : \Omega \to \mathbb{R}$ $(j = 1, \ldots, k)$ is the function to be minimized. In this paper, we consider the box-constrained case, i.e., $\Omega = \prod_{i=1}^{n}[a_j, b_j]$. Therefore, each variable vector $\mathbf{x} = (x_1, \ldots, x_n)^\intercal \in \Omega$ is such that $a_i \leq x_i \leq b_i$ for all $i \in \{1, \ldots, n\}$.

In multi-objective optimization, it is desirable to obtain a set of trade-off solutions representing the best possible compromises among the objectives (i.e., solutions such that no objective can be improved without worsening another). Therefore, in order to describe the concept of optimality in which we are interested, the following definitions are introduced [8].

Definition *1:* Let $\mathbf{x}, \mathbf{y} \in \Omega$, we say that $\mathbf{x}$ *dominates* $\mathbf{y}$ (denoted by $\mathbf{x} \prec \mathbf{y}$) with respect to the problem defined in equation (1) if and only if: *1)* $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ for all $i \in \{1, \ldots, k\}$ and *2)* $f_j(\mathbf{x}) < f_j(\mathbf{y})$ for at least one $j \in \{1, \ldots, k\}$.

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: $PS = \{\mathbf{x} \in \Omega | \mathbf{x} \text{ is a Pareto optimal solution}\}$ and its image $PF = \{\mathbf{F}(\mathbf{x}) | \mathbf{x} \in PS\})$ is called *Pareto front PF*.

As in most of multi-objective algorithms, we are interested in finding a certain (finite) number of elements of the Pareto optimal set, while maintaining a proper representation of the Pareto front.

B. MOEA/D

The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [7], 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 $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ as the well-distributed set of weights vectors, MOEA/D finds the best solution to each subproblem defined by each weights vector using the Penalty Boundary Intersection (PBI) approach [7], which has the form:

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

where

$$d_1 = \frac{\|(\mathbf{F}(\mathbf{x}) - \mathbf{z}^\star)^\intercal \mathbf{w}\|}{\|\mathbf{w}\|}$$

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

Algorithm 1: General Framework of MOEA/D

Input:
N: the number of the subproblems to be decomposed;
W: a set of weights vectors $\{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$;
T: the neighborhood size.
Output:
P: the final approximation to PS.

```
z = (z_1 = +∞, ..., z_k = +∞)^T;
Generate a random set of solutions P = {x^1, ..., x^N} in Ω;
for i = 1, ..., N do
    B_i = {i_1, ..., i_T}, such that: w^{i_1}, ..., w^{i_T} are the T closest
    weight vectors to w^i;
    z_j = min(z_j, f_j(x^i));                          // for j = 1, ..., k
while stopping criterion is not satisfied do
    for x^i ∈ P do
        Reproduction: Randomly select two indexes k, l from
        B_i, and then generate a new solution y from x^k and x^l
        by using genetic operators;
        Mutation: Apply a mutation on y to produce y';
        Update of z: z_j = min(z_j, f_j(x^i)) ;        // for j = 1, ..., k
        Update of Neighboring Solutions: For each index
        j ∈ B_i, if g(y'|w^j, z) < g(x^i|w^j, z), then set x^j = y';
return P = {x^1, ..., x^N};
```

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{z}^\star = \min\{f_i(\mathbf{x}) | \mathbf{x} \in \Omega\}$. Since $\mathbf{z}^\star = (z_1^\star, \ldots, z_k^\star)^\intercal$ is unknown, MOEA/D states each component z_j by the minimum value for each objective f_j found during the search process, for $j = 1, \ldots, k$.

In MOEA/D, a neighborhood of the weights vector $\mathbf{w}^i$ is defined as a set of its closest weights vectors in W. Therefore, the neighborhood of the i^{th} subproblem consists of all the subproblems with the weights vectors from the neighborhood of $\mathbf{w}^i$ and it is denoted by $B(\mathbf{w}^i)$.

Throughout the search process, MOEA/D finds the best solution to each subproblem maintaining a population of N solutions $P = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ where $\mathbf{x}^i \in \Omega$ is the current solution to the i^{th} subproblem. Algorithm 1 presents the general framework of MOEA/D, however, for a more detailed description, the interested reader is referred to [7].

III. Discrepancy and Low-Discrepancy Sequences

A. Discrepancy

Discrepancy theory (also called theory of distribution irregularities) is a branch of mathematics which deals with the problem of distributing points uniformly over some geometric object and evaluating the inevitably arising errors. This theory was ignited by theoretical contributions such as Weyl's equidistribution theorem [12] and Roth's theorem [13] on the irregularities of point distributions. The discrepancy of a point set, measures the nonuniformity of such points placed (without loss of generality[1]) in a unit cube $[0, 1]^s$, where $s > 0$ denotes the dimension of the unit cube. In the specialized literature there are several discrepancy measures which determine the

[1] A set of points can be normalized to a unit cube.

nonuniformity of a point set. The most widely studied discrepancy measure and commonly adopted in quasi-Monte-Carlo methods, is the well-known L_p-discrepancy, which is formally stated as follows, see [14].

Definition *4:* Let $C^s = [0,1)^s$ be a s-dimensional unit cube. Let $P = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset C^s$. For $p > 0$, the L_p-discrepancy (D_p) of P is defined by,

$$D_p(P) = \left(\int_{C^s} \left| \frac{N(P,[\mathbf{0},\mathbf{x}))}{n} - Vol\left([\mathbf{0},\mathbf{x})\right) \right|^p \delta\mathbf{x} \right)^{1/p} \tag{3}$$

where $[\mathbf{0},\mathbf{x})$ denotes the interval $[0,x_1)\times\cdots\times[0,x_s)$, $N(P,[\mathbf{0},\mathbf{x}))$ denotes the number of points of P falling in $[\mathbf{0},\mathbf{x})$, and $Vol(A)$ is the volume of A, which is the distribution function of the uniform distribution on C^s.

The smaller the value of D_p, the more uniform the distribution of points of P. With $p = \infty$, the above discrepancy is called star discrepancy ($D^\star$) and it is formally stated as follows.

Definition *5:* Let $C^s = [0,1)^s$ be a s-dimensional unit cube. Let $P = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset C^s$. The star discrepancy ($D^\star$) of P is defined by,

$$D^\star(P) = \sup_{\mathbf{x}\in C^s} \left| \frac{N(P,[\mathbf{0},\mathbf{x}))}{n} - Vol\left([\mathbf{0},\mathbf{x})\right) \right| \tag{4}$$

where $[\mathbf{0},\mathbf{x})$ denotes the interval $[0,x_1)\times\cdots\times[0,x_s)$, $N(P,[\mathbf{0},\mathbf{x}))$ denotes the number of points of P falling in $[\mathbf{0},\mathbf{x})$, and $Vol(A)$ is the volume of A.

B. Low-Discrepancy Sequences

Formally, a low-discrepancy sequence can be defined as follows.

Definition *6:* Let $P = \{\mathbf{x}_1 \ldots, \mathbf{x}_N\}$ be a sequence of points in $[0,1]^s$. P is said to be a low-discrepancy sequence if for any $N > 1$

$$D^\star(P) \leq C_s \frac{(\log N)^s}{N} \tag{5}$$

where the constant C_s depends only on the s-dimensional problem.

In the following, we present some low-discrepancy sequences that we have adopted in the study presented here.

1) Halton sequence: Let $b_1, \ldots, b_s$ be s pairwise relatively prime. The s-dimensional Halton sequence [15] of order N is defined as:

$$\mathbf{x}_n = \left(\varphi_{b_1}(n), \ldots, \varphi_{b_s}(n)\right), \qquad n = 0,1,\ldots,N-1, \tag{6}$$

where $\varphi_{b_j}(n)$ ($j = 1,\ldots,s$) is the j^{th} radical inverse function: $\varphi_{b_j}(n) = \sum_{i=0}^{l(j)} a_i(j,n) b_j^{-i-1}$. This sum is finite with the integer coefficients $a_i(j,n) \in [0, b_j - 1]$ (j and n are indexes) coming from the digit expansion of the integer n in base b_j, that is: $n = \sum_{i=0}^{l(j)} a_i(j,n) b_j^i$, where $l(j) = \lceil \log_{b_j} n \rceil$.

2) Hammersley sequence: Let $p_1, \ldots, p_{s-1}$ be the first $s-1$ prime numbers. The s-dimensional Hammersley sequence [16] of order N is defined as:

$$\mathbf{x}_n = \left(\frac{n}{N}, \varphi_{p_1}(n), \ldots, \varphi_{p_{s-1}}(n)\right), \qquad n = 0,1,\ldots,N-1, \tag{7}$$

similar to the Halton sequence, $\varphi_{p_j}(n)$ is the j^{th} radical inverse function with respect to p_j, i.e., $\varphi_{b_j}(n) = \sum_{i=0}^{l(j)} a_i(j,n) p_j^{-i-1}$, for $j = 1,\ldots,s-1$. $a_i(j,n) \in [0, p_j - 1]$ (j and n are indexes) coming from the digit expansion of the integer n in base p_j, i.e., $n = \sum_{i=0}^{l(j)} a_i(j,n) p_j^i$ and $l(j) = \lceil \log_{p_j} n \rceil$.

3) Sobol sequence: The s-dimensional Sobol sequence [17] is generated from a set of binary fractions of length w bits, $v_i^j, i = 1,\ldots,w, j = 1,\ldots,s$. The numbers v_i^j are called direction numbers. In order to generate the direction numbers for dimension j, a primitive (irreducible) polynomial on the field GF_2 with elements $\{0,1\}$ is defined. Assuming that the primitive polynomial in dimension j is given by:

$$p_j(x) = x^q + a_1 x^{q-1} + \cdots + a_{q-1}x + 1.$$

The direction numbers in dimension j are generated by using the following q-term recurrence relation

$$v_i^j = a_1 v_{i-1}^j \oplus a_2 v_{i-2}^j \oplus \cdots \oplus a_{q-1} v_{i-q+1}^j \oplus v_{i-q}^j \oplus (v_{i-q}^j / 2^q),$$

where $i > q$. $\oplus$ denotes the bitwise XOR operation. The initial numbers $v_1^j \cdot 2^w, \ldots, v_q^j 2^w$ can be arbitrary odd integers smaller than $2^1, 2^2, \ldots$, and 2^q, respectively. In this way, the Sobol sequence x_n^j ($n = \sum_{i=0}^{w} b_i 2^i, b_i \in \{0,1\}$) in dimension j is generated by

$$x_n^j = b_1 v_1^j \oplus b_2 v_2^j \oplus \cdots \oplus b_w v_w^j, \qquad n = 0,1,\ldots,N-1. \tag{8}$$

4) Faure sequence: The s-dimensional Faure sequence [18] of order N is stated as follows. Let p be the first prime numbers such that $p \geq s$ and p^m is the upper bound of the sample size. Let $c_{ij} = \binom{i}{j} \mod p, 0 \leq j \leq i \leq m$. Consider the base p representation of n, i.e., $n = \sum_{i=0}^{m-1} a_i(n) p^i$, where $a_i(n) \in [0,p)$ are integers. The first coordinate of the point $\mathbf{x}_n$ is then given by

$$x_n^1 = \sum_{j=0}^{m-1} a_j(n) p^{-j-1}, \qquad n = 0,1,\ldots,N-1. \tag{9}$$

The remaining coordinates are given by

$$\begin{cases} x_n^i = \sum_{j=0}^{m-1} a_j(n) p^{-j-1} \text{ such that:} \\ a_j(n) = \overline{a}_j(n), j \in \{0,1,\ldots,m-1\} \text{ and} \\ \overline{a}_j(n) = \sum_{l=j}^{m-1} c_{lj} a_l(n) \mod p, j \in \{0,1,\ldots,m-1\}, \end{cases} \tag{10}$$

in order $i = 2,\ldots,s$. An illustration of this and the above low-discrepancy sequence in $\mathbb{R}^2$ (i.e., with dimension $s = 2$) is presented in Fig. 1.

In the specialized literature, there exist other low-discrepancy sequences with asymptotically good behavior, i.e., with small value of C_s, such as the Niederreiter sequence [19], the Niederreiter-Xing sequence [20] or the Kronecker sequences [21]. These sequences require to solve some implementation issues, which, indeed, increase the computational

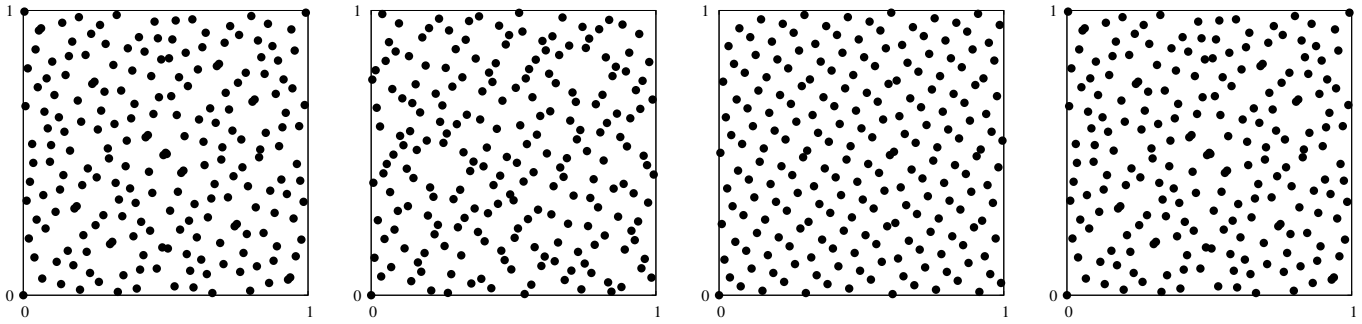

Fig. 1. Illustration of low-discrepancy sequences of order $N = 210$ in $\mathbb{R}^2$ (i.e., $s = 2$)

complexity becoming (in most cases) impractical. On the other hand, sequences with asymptotically good behavior do not necessarily perform well in practical applications, where a finite number of elements close to the beginning of the sequence is required.

IV. Constructing the Weights Vectors

One of the main problems in MOEAs that use a set of weights vectors to obtain a proper representation of the PF of a MOP (see e.g. MOEA/D [7], R2-IBEA [22], RIB-EMOA [23]), is the definition of such weights vectors. For these MOEAs, a good representation of the PF is achieved by defining a well-distributed set of weights vectors. However, as said before, when the number of objectives increases, the generation of well-distributed weights vectors becomes a problem.

In this section, we give a methodology to generate a well-distributed set of weights vectors starting from a set of points with low discrepancy.

In the following description, let us assume that k is the dimension of the weights vectors and N is the number of weights vectors to be generated.

Let $B = \{\mathbf{b}^1, \ldots, \mathbf{b}^N\}$ be the low-discrepancy sequence of order N in the $(k-1)$-dimensional unit cube, i.e., $B \subset [0,1)^{k-1}$. Each weights vector $\mathbf{w}^i \in W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$, for $i \in \{1, \ldots, N\}$, is achieved according to the transformation ($\mathbf{b} \rightarrow \mathbf{w}$) presented in Algorithm 2.

Algorithm 2: Transformation $\mathbf{b} \rightarrow \mathbf{w}$

Step 1. Sort the coefficients of each vector $\mathbf{b}^i \in B$ such that: $0 \leq b^i_1 \leq \cdots \leq b^i_{k-1}$, for each $i \in \{1, \ldots, N\}$.
Step 2. Set the weights vector $\mathbf{w}^i = (w^i_1 \ldots, w^i_k)^\intercal$ as:

$$w^i_j = \begin{cases} b^i_1, & \text{if } j = 1 \\ b^i_j - b^i_{j-1}, & \text{if } 1 < j < k \\ 1 - b^i_{j-1}, & \text{if } j = k \end{cases} \tag{11}$$

Since it is possible to recover the sorted b_j's by means of the partial sums of the w_j's, the transformation $\mathbf{b} \rightarrow \mathbf{w}$ is $(n-1)!$ to 1. In particular, a set of weights vectors denotes a $(k-1)$-simplex in $\mathbb{R}^k$. Because *i)* each swap in a sort is a linear transformation, *ii)* the transformation formula (equation (11)) is linear, and *iii)* linear transformations preserve uniformity of distributions, the uniformity of $\mathbf{b}$ implies the uniformity of $\mathbf{w}$ on the $(k-1)$-simplex. Therefore, considering that B is built from a low-discrepancy sequence, the low discrepancy of B implies the low discrepancy of W. The computational complexity of this transformation over all the points in B is given by $O(N \times k \times \log k)$.

Particularly, the low-discrepancy sequences presented in Section III-B, have the property of generating the vector $\mathbf{b} = \mathbf{0}$, i.e., the origin point. Therefore, in this specific case, the transformation $\mathbf{b} \rightarrow \mathbf{w}$ shall generate the weights vector $(0, 0, \ldots, 0, 1)^\intercal$, i.e., the k^{th} canonical basis in $\mathbb{R}^k$. An illustration of different weights vectors generated by the transformation $\mathbf{b} \rightarrow \mathbf{w}$ and using different low-discrepancy sequences is shown in Fig. 2.

It is worth noting that a low discrepancy of W does not imply the points in W are symmetrically stated. However, a better symmetry of points can be reached by performing all left shifts with feedback at each point $\mathbf{w} \in W$ and selecting (from W and all the shifts) the best well-spread points. This transformation (denoted as $W \rightarrow W^\star$) is described in Algorithm 3.

Since the simple transformation $\mathbf{b} \rightarrow \mathbf{w}$ generates the k^{th} canonical basis in $\mathbb{R}^k$, **Step 2** (in Algorithm 3) shall generate all possible left shifts with feedback, i.e., it shall generate all canonical basis in R^k. For example, for the weights vector $(0, 0, 1)^\intercal$, all possible left shifts with feedback are: $(0, 1, 0)^\intercal$ and $(1, 0, 0)^\intercal$, whereas for $(0.2, 0.3, 0.5)^\intercal$ the left shifts generate: $(0.3, 0.5, 0.2)^\intercal$ and $(0.5, 0.2, 0.3)^\intercal$. In Algorithm 3, W^s stores both the original weights vectors W and the shifts with feedback of each element in W.

The selection of well-distributed points is carried out in **Step 3** of Algorithm 3. Initially, the central vector $\mathbf{w}^c = (1/k, \ldots, 1/k)^\intercal$ is stored in the final set of weights $W^\star$. Then, the distances between the central vector and each weights vector in W^s are computed. In this way, while the cardinality of $W^\star$ is less than the number of required weights (N), the weights vector in W^s farthest to the set of final weights $W^\star$ is removed and stored in $W^\star$. In this way, the set $W^\star$ shall contain *1)* the canonical basis in $\mathbb{R}^k$ and *2)* the central vector $\mathbf{w}^c$. Fig. 3 shows the transformation $W \rightarrow W^\star$ for a set of weights vectors generated by the Sobol sequence of order

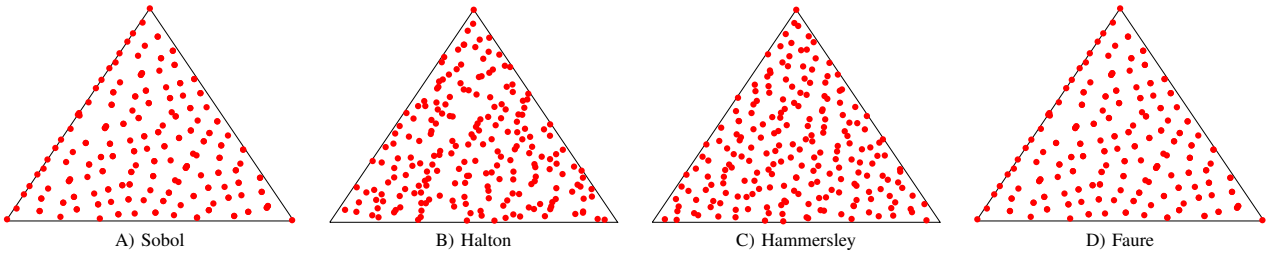

Fig. 2. Illustration of weights vectors generated by the transformation $\mathbf{b} \rightarrow \mathbf{w}$ using different low-discrepancy sequences of order $N = 210$

Algorithm 3: Transformation $W \rightarrow W^\star$

Step 1. Set $W^s = W$

Step 2. Let $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^{|W|}\}$

for $j \in \{1, \ldots, |W|\}$ **do**

– compute all left shifts with feedback of $\mathbf{w}^j \in W$ and add them to W^s

end for

Step 3. Let $W^s = \{\mathbf{w}^1, \ldots, \mathbf{w}^{|W^s|}\}$ and $\mathbf{w}^c = (1/k, \ldots, 1/k)^\intercal$ then

- set $W^\star = \{\mathbf{w}^c\}$
- **for** $j \in \{1, \ldots, |W^s|\}$ **do**
  - compute $d_j = ED(\mathbf{w}^j, \mathbf{w}^c)$
  - set $v_j = AVAILABLE$

  end for
- **while** $|W^\star| < N$ **do**

  $k = \underset{j \mid v_j = AVAILABLE}{\arg\max} \; d_j$

  $v_k = UNAVAILABLE$

  $W^\star = W^\star \cup \{\mathbf{w}^k\}$

  for $l \in \{1, \ldots, |W^s|\}$ **do**

  $d_l = \min(ED(\mathbf{w}^l, \mathbf{w}^k), d_l)$, iff $v_l = AVAILABLE$.

  end for

  end while

where $ED(\mathbf{a}, \mathbf{b})$ denotes the Euclidean distance between $\mathbf{a}$ and $\mathbf{b}$.

$N = 210$.

Since the transformation $W \rightarrow W^\star$ considers the left shifts with feedback and the set W by itself, the cardinality of W^s becomes $k \times |W|$. The computational complexity of transformation $W \rightarrow W^\star$, is stated in **Step 3**, which is given by $O(N \times |W^s|) = O(N \times k \times |W|)$. Since cardinality of W is N, the final complexity is $O(N^2 \times k)$.

It is worth noting that the distribution of points can be improved by computing the permutations of each weights vector in W instead of computing the shifts with feedback. Note however that the cardinality of W^s shall become $k! \times |W|$, which indeed, increases the computational complexity of the transformation $W \rightarrow W^\star$ to $O(N^2 \times k!)$. In this study, we are interested in the design of a practical mechanism to generate a well-spread set of weights vectors. For this reason, we limit this work to the use of left shifts with feedback.

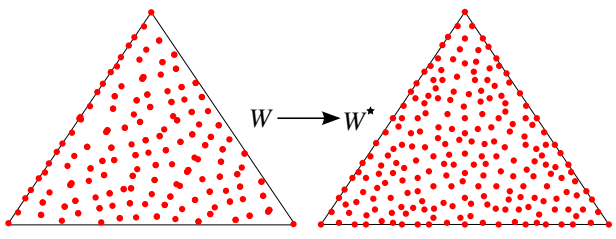

Fig. 3. Illustration of the transformation $W \rightarrow W^\star$ for a set of weights W generated by the Sobol sequence of order $N = 210$

V. Experimental Analysis

A. Performance assessment

In this study, we are interested in the good distribution of weights vectors and their impact on the search process of MOEA/D [7], in high-dimensional objective spaces. In the specialized literature, there are several methods that have been used to assess the diversity of nondominated points—see e.g., the performance measures spacing [24], spread [25] and entropy [26]. Since the weights set denotes a set of nondominated points, one of above mentioned measures could be employed to assess the distribution of the weights set. However, as it was discussed in [27], [28], in some cases, these measures don't provide a correct assessment of the distribution. Not only does the Hypervolume indicator (I_H) [29] assess the approximation of points to the real Pareto front, but it also assesses the distribution of points along the Pareto front. However, in some cases, this metric does not necessarily assess a good diversity of points along the Pareto front. When the Pareto front describes a convex surface, I_H benefits to samplings with more solutions close to the convex part. Conversely, if the Pareto front describes a concave surface, I_H favors samplings with more solutions in the extreme portions of the Pareto front. Since the weights set denotes a hyperplane, neither a convexity nor a concavity is described. It means that I_H shall assess in a proper way the distribution of points in the hyperplane. Therefore, in this study, we assess the distribution of weights vectors by using the Hypervolume indicator. To compute I_H of the weights set, the reference point was stated as the unitary vector, i.e., $(1, \ldots, 1)^\intercal$. With this, we can assess a good distribution to the extreme portions of the $(k-1)$-simplex.

As we said before, simplex-lattice design [9], is perhaps, the best strategy used in the construction of evenly distributed weights vectors. However, when the dimension of the weights increases, this methodology becomes impractical. A strategy to deal with high dimensional spaces is the one proposed in [30] (called herein two-layered simplex-lattice design). This strategy uses the simplex-lattice design to generate an outside layer and an inside layer in the weights set. Fig. 4 illustrates

TABLE I
CONFIGURATION FOR THE TWO-LAYERED SIMPLEX-LATTICE DESIGN

Dimension	Layers	Layer configuration	Number of weights
3	1	$H = 19$	210
4	1	$H = 9$	220
5	1	$H = 6$	210
6	2	$H_1 = 4, H_2 = 3$	182
7	2	$H_1 = 4, H_2 = 2$	238
8	2	$H_1 = 3, H_2 = 2$	156
9	2	$H_1 = 3, H_2 = 2$	210
10	2	$H_1 = 3, H_2 = 2$	275
11	2	$H_1 = 2, H_2 = 2$	132
12	2	$H_1 = 2, H_2 = 2$	156
13	2	$H_1 = 2, H_2 = 2$	182
14	2	$H_1 = 2, H_2 = 1$	119
15	2	$H_1 = 2, H_2 = 2$	135

the two-layered simplex-lattice design in $\mathbb{R}^3$ by using $H_1 = 2$ (for the outside layer) and $H_2 = 1$ (for the inside layer).

In this study, we compare the distribution of weights of our proposed approach with respect to the two-layered simplex-lattice design. The complete configuration of H values for different dimensions of the two-layered simplex-lattice design is shown in Table I. Note however, that we only use the two-layer strategy when the dimensionality is greater than 5; otherwise, a single layer is employed.

B. On the low-discrepancy sequences

As we said before, the set of weights vectors generated by different low-discrepancy sequences is compared with respect to the simplex-lattice design. Since the number of weights in simplex-lattice design depends on the configuration of the H values (even for the two-layer strategy), we adopted the same number of weights vectors for our proposed methodology.

Table II shows the I_H indicator for the weights vectors generated by the different strategies compared in this study. For an easy interpretation, the best values are in **boldface** and the second best values are underlined. From this table, we can see that, in most cases, the weights set generated by the simplex-lattice design achieved a higher I_H value than any other strategy. This is not surprising, since as mentioned before, the simplex-lattice design is probably the best strategy to generate equally distributed weights vectors. However, when the dimensionality of the weights increases, the performance of the two-layered simplex-lattice strategy starts to deteriorate.

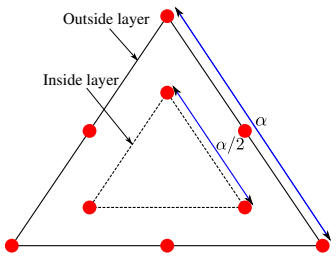

Fig. 4. Illustration of the two-layered simplex-lattice design. The outside layer is stated by $H_1 = 2$ (generating six weights vectors), while the inside layer is set by $H_2 = 1$ (generating three weights vectors)

This is due mainly to the fact that the two-layer configuration does not provide a sufficiently good distribution along the $(k-1)$-simplex. In contrast, the use of low-discrepancy sequences is not limited for any configuration. As we can see from Table II, when the dimensionality of the weights increases ($k > 10$) the proposed methodology becomes better than the simplex-lattice design. In particular, from the low-discrepancy sequence, the Sobol's sequence became the best choice to generate the set of weights vectors. An example of different weights vectors in $\mathbb{R}^3$ generated by different methods is shown in Fig. 5. To contrast the proposed methodology, we include the set of weights vectors generated by simplex-lattice design (Fig. 5-E) and a sampling of random weights (Fig. 5-F).

C. On the performance of the weights vectors in MOEA/D

In order to validate the impact of different weights vectors in MOEA/D, we adopted three MOPs taken from the literature: The well-known DTLZ1 and DTLZ2 test problems [31] and a rotated version of DTLZ2 (DTLZ2_ROT) [30]. These problems have connected Pareto fronts, which describe a linear, a concave and a convex surface, respectively. Such problems are controlled by the parameter K which states the difficulty of each problem (for more details see [31]). Since we only test the distribution of points produced by MOEA/D, we state K with the minimum allowable value (the smaller the value, the problem is easier to solve). For each set of weights

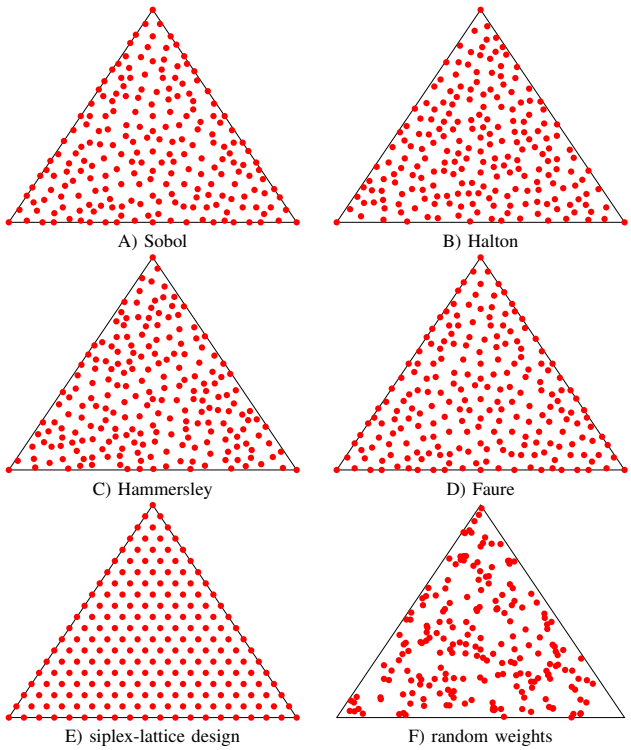

Fig. 5. Illustration of weights vectors in $\mathbb{R}^3$ generated by different strategies. Figures A to D correspond to the weights vectors generated by low-discrepancy sequences of order $N = 210$. Figures E and F correspond to the weights vectors generated by the simplex-lattice design (with only one layer $H = 19$, i.e., 210 weights vectors) and 210 random weights vectors, respectively.

TABLE II
HYPERVOLUME FOR THE WEIGHTS SET OBTAINED BY DIFFERENT METHODS.

Dimension	Simplex-lattice	Sobol	Halton	Hammersley	Faure	Random
3	**0.806094**	0.805209	0.797167	0.797505	0.805209	0.784856
4	**0.924554**	0.920339	0.905950	0.903775	0.915613	0.895743
5	**0.967593**	0.959349	0.945150	0.944709	0.954705	0.930800
6	**0.983109**	0.975670	0.954625	0.951910	0.968713	0.933604
7	**0.993778**	0.988688	0.967964	0.965749	0.982565	0.954102
8	**0.994831**	0.991509	0.949837	0.963820	0.984827	0.945335
9	**0.997901**	0.995257	0.967757	0.972456	0.989662	0.956424
10	**0.999161**	0.997850	0.980030	0.976338	0.993499	0.965295
11	0.996380	**0.996909**	0.956934	0.959910	0.994217	0.947044
12	0.998036	**0.998771**	0.961059	0.965120	0.996590	0.953036
13	0.998941	**0.999209**	0.957777	0.968477	0.997847	0.954429
14	0.999387	**0.999273**	0.957909	0.958817	0.995907	0.950421
15	**0.999674**	0.999616	0.961711	0.959053	0.998153	0.952001

and each MOP, MOEA/D is performing the same number of generations (3,000). For all the instances, we employed the PBI approach and the parameters for MOEA/D are set as in [7]. However, the number of subproblems (N) is stated by the number of weights vectors defined by the simplex-lattice design (see Table I). In Table III, we show the I_H values achieved by the nondominated solutions obtained by MOEA/D using different weights sets. These values were computed using the reference vector $(1.1,\ldots,1.1)^\intercal$ for DTLZ2 and DTLZ2_ROT and $(0.6,\ldots,0.6)^\intercal$ for DTLZ1. For an easy interpretation, the best values are in **boldface** while the second best values are underlined.

From this table, we can see that for DTLZ1, MOEA/D obtained a better I_H value using the weights vectors generated by the simplex-lattice design in the majority of the dimensions. This performance was expected since the original weights vectors generated by the simplex-lattice design (which indeed, describes a hyperplane) obtained a better I_H value (see Table II). Note however, that when the number of objectives increases, this set of weights deteriorates the performance of MOEA/D. For DTLZ2 (which has a concave Pareto front), the performance of MOEA/D using the weights vectors generated by the simplex-lattice design became very similar. The I_H indicator showed a better performance using these weights vectors than the one generated by the low-discrepancy sequences. However, analogously to DTLZ1, the performance of MOEA/D decayed as the number of objectives increases (see e.g. for more than 10 objectives). The main reason of this is that the configuration of layers stated in Table I was not suitable to cover the whole objective space. Note however that another configuration of layers could result in a large number of weights vectors, which can become impractical. The most contrasting results were in DTLZ2_ROT. For this particular problem, the performance of MOEA/D using the weights vectors generated by the low-discrepancy sequences became better than the one generated by the simplex-lattice design.

It is worth noting that the two-layer strategy, indeed, favored the samplings with solutions towards the extreme portions of the Pareto front. For this reason, the use of these weights vectors benefits the search of MOEA/D in DTLZ2. However, for DTLZ2_ROT, the performance of MOEA/D became poor. In contrast, the performance of MOEA/D using the weights vectors generated by the low-discrepancy sequences became better than the one achieved by using the simplex-lattice design in DTLZ2_ROT. In general terms, for high dimensional objective spaces, the performance of weights vectors generated by low-discrepancy sequences outperformed the one achieved by the simplex-lattice design.

VI. CONCLUSIONS

We have presented a methodology for the construction of weights vectors. The proposed approach is based on a finite set of points with low discrepancy. Such points are projected to the $(k-1)$-simplex (which defines the set of weights vectors). The discrepancy of such points is reflected in the resulting weights set. Therefore, the lower the discrepancy, the better the distribution of points in the $(k-1)$-simplex. To validate our proposed methodology, we assessed the distribution of points by using the Hypervolume indicator. Our proposed approach was compared with respect to the simplex-lattice design and (for more than five objectives) to the two-layer strategy. We showed that our proposed approach produces a set of weights vectors with a proper distribution. Our experiments showed that our proposed approach became effective when the dimensionality of the weights increases.

Given the characteristics of our proposed approach, we have shown the flexibility of the method by not depending on any parameter for the construction of weights as occurs with other methodologies—we only need to define the dimensionality and the number of weights vectors required. Regarding efficiency, our proposed approach achieved a computational complexity of $O(N^2 \times k)$, which is comparable to one iteration of traditional MOEAs reported in the specialized literature, see [32]. Therefore, the study presented herein, suggests that the proposed methodology is appropriate for the generation of weight vectors in high-dimensional objective spaces.

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.

TABLE III

I_H INDICATOR OF THE NONDOMINATED SOLUTIONS OBTAINED BY MOEA/D USING DIFFERENT WEIGHTS VECTORS FOR DTLZ1, DTLZ2 AND DTLZ2_ROT

MOP	Obj.	Simplex-lattice	Sobol	Halton	Hammersley	Faure
DTLZ1	3	**0.191762**	0.191651	0.190646	0.190688	0.191651
	4	**0.124885**	0.124621	0.123722	0.123586	0.124326
	5	**0.076747**	0.076490	0.076046	0.076032	0.076345
	6	**0.046392**	0.046276	0.045947	0.045905	0.046167
	7	**0.027945**	0.027905	0.027743	0.027726	0.027857
	8	**0.016776**	0.016763	0.016600	0.016655	0.016737
	9	**0.010074**	0.010068	0.010015	0.010024	0.010058
	10	**0.006046**	0.006045	0.006027	0.006024	0.006040
	11	0.003626	**0.003626**	0.003607	0.003608	0.003625
	12	0.002176	**0.002176**	0.002167	0.002168	0.002176
	13	0.001306	**0.001306**	0.001301	0.001302	0.001306
	14	0.000784	**0.000784**	0.000781	0.000781	0.000783
	15	**0.000470**	0.000470	0.000469	0.000469	0.000470
DTLZ2	3	**0.767120**	0.763886	0.759032	0.757787	0.763886
	4	**1.058964**	1.044281	1.030938	1.024972	1.039388
	5	**1.308739**	1.274996	1.263712	1.254561	1.263746
	6	**1.525514**	1.489634	1.454719	1.449275	1.457549
	7	**1.779060**	1.725981	1.650556	1.648354	1.690680
	8	**1.975558**	1.906824	1.753068	1.780832	1.803935
	9	**2.233472**	2.142196	1.990097	2.015975	2.032233
	10	**2.494236**	2.418884	2.261698	2.241009	2.254849
	11	2.568114	**2.587977**	2.336576	2.315215	2.337185
	12	2.844325	**2.923489**	2.579982	2.627505	2.589630
	13	3.143586	**3.225883**	2.841030	2.852258	2.905227
	14	3.399125	**3.491341**	3.037364	3.031167	2.830027
	15	3.743584	**3.880167**	3.330654	3.332258	3.245899
DTLZ2_ROT	3	**1.277146**	1.277078	1.273771	1.274244	1.277078
	4	**1.447256**	1.446665	1.443243	1.442951	1.440925
	5	1.599976	**1.600294**	1.597297	1.597032	1.593776
	6	1.763764	**1.764475**	1.761386	1.761887	1.758262
	7	1.936711	**1.942841**	1.942691	1.941709	1.934759
	8	2.096606	**2.138559**	2.134089	2.137820	2.123975
	9	2.296420	**2.353541**	2.352577	2.351998	2.350841
	10	2.517679	**2.589707**	2.589346	2.589701	2.588473
	11	2.707060	**2.847970**	2.846558	2.845522	2.843940
	12	2.989443	**3.130805**	3.130644	3.130752	3.129842
	13	3.299406	**3.446850**	3.445989	3.446519	3.442939
	14	3.624428	**3.780643**	3.768784	3.780344	3.608938
	15	3.998117	**4.159963**	4.128397	4.158430	4.152579

[2] V. Pareto, *Cours d'Economie Politique* . Lausanne: F. Rouge, 1896.

[3] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, Eds., Athens, Greece, 2001, pp. 95–100.

[4] 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.

[5] J. Knowles and D. Corne, "Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 100–116, April 2003.

[6] J. Horn, "Multicriterion Decision Making," in *Handbook of Evolutionary Computation*, T. Bäck, D. Fogel, and Z. Michalewicz, Eds. IOP Publishing Ltd. and Oxford University Press, 1997, vol. 1, pp. F1.9:1 – F1.9:15.

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

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

[9] H. Scheffé, "Experiments With Mixtures," *Journal of the Royal Statistical Society*, vol. Series B (Methodological), 20, no. 2, pp. 344–360, 1958.

[10] ——, "The Simplex-Centroid Design for Experiments with Mixtures," *Journal of the Royal Statistical Society*, vol. Series B (Methodological), 25, no. 2, pp. 235–263, 1963.

[11] J. A. Cornell, "Some Comments on Designs for Cox's Mixture Polynomial," *Technometrics*, vol. 17, no. 1, pp. 25–35, 1975.

[12] P. Bohl, "ber ein in der theorie der skutaren strungen vorkommendes problem," *J. reine angew. Math.*, vol. 135, pp. 189–283, 1909.

[13] K. F. Roth, "Rational approximations to algebraic numbers," *Mathematika*, vol. 2, pp. 1–20, June 1955.

[14] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, ser. CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Ed. Philadelphia: Society for Industrial and Applied Mathematics, 1992.

[15] J. H. Halton, "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, vol. 2, pp. 84–90, December 1960.

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

[17] I. M. Sobol, "On the distribution of popoint in a cube and the approximate evaluation integrals," *U.S.S.R. Computational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112, 1967.

[18] H. Faure, "Discrepance de suites associees a un systeme de numeration (en dimension s)," *Acta Arithmetica*, vol. 53, pp. 337–351, 1982.

[19] H. Niederreiter, "Low-discrapancy and low-dispersion dsequences," *Journal of Number Theory*, vol. 30, pp. 51–70, 1988.

[20] H. Niederreiter and C. Xing, "Low-discrapancy sequences and global ffunction fields with many rational places," *Finite Fields and Their Applications*, vol. 2, pp. 241–273, 1996.

[21] G. Larcher, "On the distribution of s-dimensional kronecker-sequences," *Acta Arithmetica*, vol. 51, no. 4, pp. 335–347, 1988.

[22] D. H. Phan and J. Suzuki, "R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization," in *CEC'2013*. IEEE, 2013, pp. 1836–1845.

[23] S. Zapotecas Martínez, V. A. Sosa Hernández, H. E. Aguirre, K. Tanaka, and C. A. C. Coello, "Using a Family of Curves to Approximate the Pareto Front of a Multi-Objective Optimization Problem," in *PPSN XIII*. Ljubljana, Slovenia: Springer, September 2014, pp. 682–691.

[24] J. R. Schott, "Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization," Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[25] M. Li and J. Zheng, "Spread assessment for evolutionary multi-objective optimization," in *Evolutionary Multi-Criterion Optimization*. Springer Berlin Heidelberg, 2009, vol. 5467, pp. 216–230.

[26] A. Farhang-Mehr and S. Azarm, "Diversity Assessment of Pareto Optimal Solution Sets: An Entropy Approach," in *Congress on Evolutionary Computation (CEC'2002)*, vol. 1. Piscataway, New Jersey: IEEE Service Center, May 2002, pp. 723–728.

[27] J. Yan, C. Li, Z. Wang, L. Deng, and D. Sun, "Diversity metrics in multi-objective optimization: Review and perspective," in *Integration Technology, 2007. ICIT '07. IEEE International Conference on*, March 2007, pp. 553–557.

[28] M. Li, S. Yang, and X. Liu, "Diversity comparison of pareto front approximations in many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2568–2584, Dec 2014.

[29] E. Zitzler and L. Thiele, "Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Case Study," in *Parallel Problem Solving from Nature V*, A. E. Eiben, Ed. Amsterdam: Springer-Verlag, September 1998, pp. 292–301.

[30] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 4, pp. 577–601, Aug 2014.

[31] 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. Ajith *et al.*, Eds. USA: Springer, 2005, pp. 105–145.

[32] C. A. Coello Coello, "A Comprehensive Survey of Evolutionary-Based Multiobjective Optimization Techniques," *Knowledge and Information Systems. An International Journal*, vol. 1, no. 3, pp. 269–308, Agosto 1999.