Evolutionary Many-objective Optimization based on Kuhn-Munkres' Algorithm

José A. Molinet Berenguer and Carlos A. Coello Coello [⋆]

Computer Science Department,CINVESTAV-IPN
Av. IPN 2508. San Pedro Zacatenco, México D.F. 07300, MÉXICO
jmolinet@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx

Abstract. In this paper, we propose a new multi-objective evolutionary algorithm (MOEA), which transforms a multi-objective optimization problem into a linear assignment problem using a set of weight vectors uniformly scattered. Our approach adopts uniform design to obtain the set of weights and Kuhn-Munkres' (Hungarian) algorithm to solve the assignment problem. Differential evolution is used as our search engine, giving rise to the so-called Hungarian Differential Evolution algorithm (HDE). Our proposed approach is compared with respect to a MOEA based on decomposition (MOEA/D) and with respect to an indicator-based MOEA (the *S metric selection Evolutionary Multi-Objective Algorithm*, SMS-EMOA) using several test problems (taken from the specialized literature) having from two to ten objective functions. Our preliminary experimental results indicate that our proposed HDE outperforms MOEA/D and is competitive with respect to SMS-EMOA, but at a significantly lower computational cost.

Keywords: Many-objective optimization, Multi-objective evolutioanry algorithms, Kuhn-Munkres algorithm

1 Introduction

A large number of problems that arise in academic and industrial areas have several conflicting objectives that need to be optimized simultaneously [7]; they are called multi-objective optimization problems (MOPs). The most commonly adopted notion of optimum in multi-objective optimization is Pareto optimality, which refers to finding the best possible trade-offs among the objectives of a multi-objective problem. These trade-off solutions constitute the so-called Pareto optimal set. The image of the Pareto optimal set is called the Pareto front. Among the different techniques available to solve MOPs, multi-objective evolutionary algorithms (MOEAs) have become very popular, mainly because of their flexibility and ease of use. Modern MOEAs normally aim at producing, in a single run, several different solutions, which are as close as possible to the true Pareto front [7]. For several years, MOEAs adopted a selection mechanism based

[⋆] The second author acknowledges support from CONACyT project no. 221551.

on Pareto optimality. However, in recent years, it was found that Pareto-based MOEAs cannot properly differentiate individuals when dealing with problems having four or more objectives (the so-called many-objective optimization problems [13]). This has motivated the development of alternative selection schemes from which the use of performance indicators has been (until now) the most popular choice [26]. When using indicator-based selection, the idea is to identify the solutions that contribute the most to the improvement of the performance indicator adopted in the selection mechanism.

From the several performance indicators currently available, the hypervolume [24] has become the most popular choice for implementing indicator-based MOEAs, mainly because of its good theoretical properties [5]. The hypervolume is the only unary indicator that is known to be Pareto compliant and it has been proved that its maximization is equivalent to finding the Pareto optimal set [11]. However, the main disadvantage of adopting this indicator is that the best algorithms known to compute the hypervolume have a computational cost which grows exponentially on the number of objectives [4]. Although some researchers have proposed schemes to approximate the hypervolume contributions at an affordable computational cost (see for example [1]), the performance of such approaches seems to degrade very quickly in high dimensionality at the expense of reducing their computational cost. This has motivated the development of other selection schemes based on different performance indicators (see for example [6]).

On the other hand, MOEAs based on decomposition have also become popular in recent years. Perhaps, MOEA/D is the most popular MOEA based on decomposition. This algorithm decomposes the MOP into N scalar optimization subproblems and it solves these subproblems simultaneously using an evolutionary algorithm. MOEA/D has shown to be a good alternative to solve MOPs with low or high dimensionality (regarding objetive function space). However, MOEA/D has two important disadvantages. The first is that it generates a new solution from an unique neighborhood, i.e., the new solution cannot be generated from individuals of different neighborhoods. And, the second is that a new solution with a high fitness can replace several solutions, and then, the population can lose diversity, see Figure 1. Li and Zhang proposed in [16] a variant of MOEA/D and they called it "MOEA/D-DE". This proposal allows that a new individual will be generated from individuals of different neighborhoods. Also, it restricts the number of solutions that can be replaced by the same individual. However, both proposals MOEA/D and MOEA/D-DE generate a new solution, and then, they look in which subproblem the new solution is better than the current solution but they do not consider the case where the solution which was replaced could improve the solution of another subproblem, i.e, both algorithms assign the best individual to each subproblem in an independent way, without considering the best assignment globally. Figure 2 shows the assignment made by MOEA/D and MOEA/D-DE and Figure 3 shows the global optimal assignment.

In this paper, we propose the use of an approach that is conceptually closer to MOEA/D, but that, instead of doing a scalarization, it transforms the original

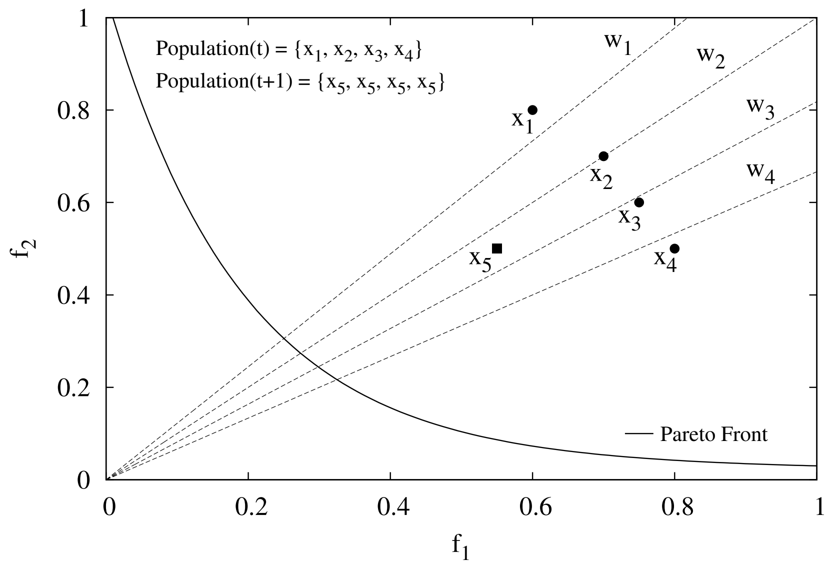

Fig. 1: Disadvantage of MOEA/D when it replaces the solutions. For each weight vector w_i, $i = 1, 2, 3, 4$, the solution x_5 has the highest utility value. Therefore, x_5 is the best solution of the four subproblems.

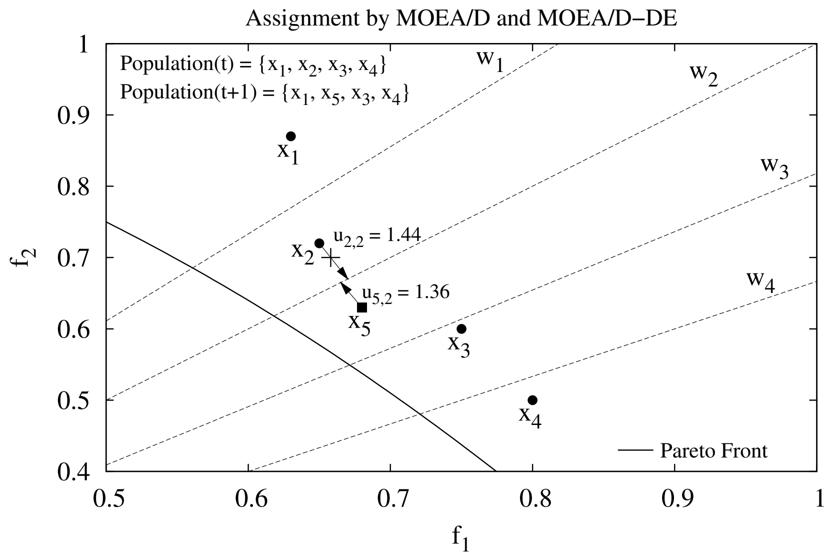

Fig. 2: The new solution x_5 is assigned to the subproblem w_2, therefore, the solution x_2 is replaced by x_5. It is important to note that the solution x_2 is better than solution x_1 for the subproblem w_1. However, MOEA/D and MOEA/D-DE eliminate x_2.

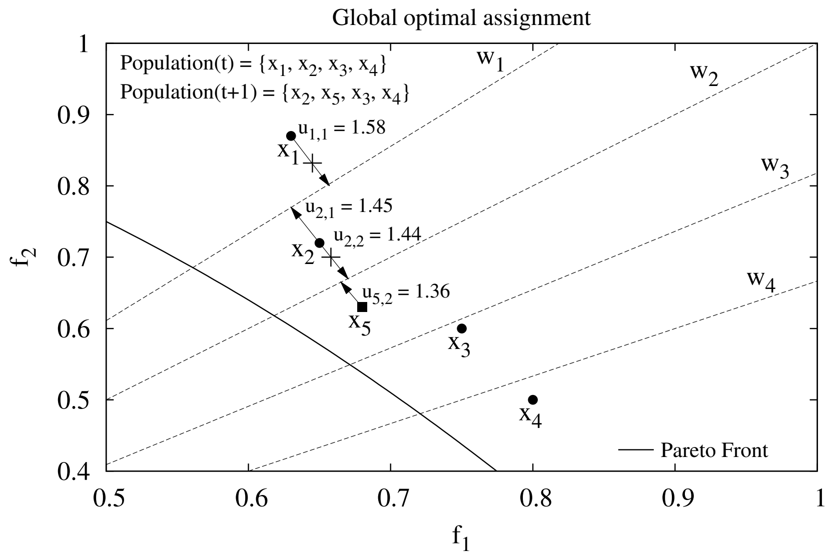

Fig. 3: The new solution x_5 is assigned to the subproblem w_2 and the solution x_2 is assigned to the subproblem w_1. Therefore, the solution x_1 is eliminated.

MOP into an assignment problem. Uniform design is adopted to obtain the set of weights, and the Kuhn-Munkres (Hungarian) algorithm [15] is used to solve the resulting assignment problem. The search engine of our proposed approach is differential evolution [19], which has been found to be a competitive search engine for single-objective optimization. As we will see later on, our results indicate that our proposed approach is very promising, particularly for solving many-objective optimization problems.

The remainder of this paper is organized as follows. The Kuhn-Munkres algorithm is described in Section 2 and in Section 3 we describe in detail our proposed approach. The experiments performed and the results obtained are described and discussed in Section 4. Finally, our conclusions and some possible paths for future work are briefly discussed in Section 5.

2 Kuhn-Munkres Algorithm

The matching or assignment problem is a fundamental class of combinatorial optimization problems. In its most general form, an assignment problem can be stated as follows: a number n of agents and a number m of tasks are given, possibly with some restrictions on which agents can perform each particular task. A cost is incurred for each agent performing some task, and the goal is to perform all tasks in such a way that the total cost of the assignment is minimized [15]. The Linear Assignment Problem (LAP) is the simplest of the assignment problems. In the canonical LAP, the number of agents and tasks is the same, and any agent can be assigned to perform any task. Formally, LAP can be formulated as follows.

Definition 1. *Given a set of agents $A = \{a_1, ..., a_n\}$, a set with the same number of tasks $T = \{t_1, ..., t_n\}$ and the cost function $C : A \times T \to \mathbb{R}$, and let $\Phi : A \to T$ the set of all possible bijections between A and T*

$$\underset{\phi \in \Phi}{\text{minimize}} \sum_{a \in A} C(a, \phi(a)) \tag{1}$$

Usually, the cost function is also viewed as a squared real-valued matrix C with elements $C_{ij} = C(a_i, t_j)$, and the set Φ of all possible bijections between A and T as a set of assignment matrices $\mathcal{X}$. The LAP can be expressed as an integer linear program:

$$\begin{aligned}
\underset{x \in \mathcal{X}}{\text{minimize}} \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_{ij} \\
\text{subject to:} \quad & \sum_{i=1}^{n} x_{ij} = 1, \ \forall j \in \{1, .., n\}, \\
& \sum_{j=1}^{n} x_{ij} \leq 1, \ \forall i \in \{1, ..., n\}, \\
& x_{ij} \in \{0, 1\}, \ \forall i, j \in \{1, ..., n\}
\end{aligned} \tag{2}$$

In 1955, Harold W. Kuhn [15] proposed an algorithm for constructing a maximum weight perfect matching in a bipartite graph. His pioneering work in this area, is a combinatorial optimization algorithm that solves the assignment problem in polynomial time. Kuhn explained how the works of two Hungarian mathematicians, D. König and E. Egerváry, had contributed to the invention of his algorithm, which is the reason why he called it the *Hungarian Method*. James Munkres [17] reviewed Kuhn's work in 1957 and made several important contributions to the theoretical aspects of the algorithm. Munkres found that the algorithm is (strongly) polynomial and proposed an improved version of $O(n^3)$. The contribution of Munkres to the development of the Hungarian algorithm has led to the algorithm which is being referred to as the Kuhn-Munkres algorithm. An extension of this algorithm for rectangular matrices was introduced by Bourgeois and Lassalle in 1971 [3]. The extension to rectangular matrices allows the algorithm to operate in assignment problems where the numbers of agents and tasks are unequal.

3 Our proposed approach

We propose here an alternative selection mechanism for MOEAs which is not based on Pareto dominance or on any performance indicator. The main motivation of this work is to avoid the scalability problems of Pareto-based selection schemes as well as the excessive computational cost of adopting the hypervolume contribution for selecting solutions. The algorithm presented here transforms the selection process into a linear assignment problem, which is solved using Kuhn-Munkres algorithm. As we will see, the solution of this LAP allows convergence towards the true Pareto front and, at the same time, a good distribution of solutions along the Pareto front. The proposed MOEA adopts the recombination operators of differential evolution to create new individuals at each generation and the Hungarian algorithm in its selection scheme. Because of this, our proposed approach is called *Hungarian Differential Evolution* (HDE).

At each g^{th} generation of the HDE algorithm we have a parent population P_g of n individuals and a population P_g^* of n offspring obtained from P_g. Let $Q_g = P_g \cup P_g^*$ be the set of $2n$ solutions in the g^{th} generation. Then, a linear assignment problem is created using the k-dimensional objective vectors from Q_g and n weight vectors uniformly spread in objective function space. In the context of a selection mechanism for MOEAs, a LAP can be understood as follows: we have $2n$ individuals and n vectors well-distributed in the $(k-1)$-dimensional unit simplex of the objective space. A cost is incurred for each individual representing some vector in the Pareto Front approximation. The goal is to describe all regions covered by the n vectors using only n individuals in such a way that the total cost of the assignment is minimized. The main task is how to construct a cost matrix such that it minimizes the total cost involved in retaining the solutions which are a good approximation of the Pareto Front. This procedure is described next.

First, the $2n$ vectors of objective values in Q_g are normalized to reduce the current objective space to a unit hypercube, so that we can deal with non-commensurable objective functions. The maximum $\boldsymbol{z}^{max}$ and minimal $\boldsymbol{z}^{min}$ vectors are calculated for this purpose.

$$\begin{aligned} \boldsymbol{z}^{max} &= [z_1^{max}, ..., z_k^{max}]^T, & z_i^{max} &= \max_{j=1,\dots,2n} f_i(\boldsymbol{x}_j), \quad i = 1, ..., k, \\ \boldsymbol{z}^{min} &= [z_1^{min}, ..., z_k^{min}]^T, & z_i^{min} &= \min_{j=1,\dots,2n} f_i(\boldsymbol{x}_j), \quad i = 1, ..., k, \end{aligned} \tag{3}$$

where $f_i(\boldsymbol{x}_j)$ is the i^{th} objective value of the j^{th} individual in Q_g, and its normalized value $f_i(\boldsymbol{x}_j)$ is calculated as:

$$\tilde{f}_i(\boldsymbol{x}_j) = \frac{f_i(\boldsymbol{x}_j) - z_i^{min}}{z_i^{max} - z_i^{min}}, \quad j = 1, ..., 2n, \ i = 1, ..., k. \tag{4}$$

Let W be a set of n weight vectors uniformly scattered in objective space.

$$W \subset \mathbb{W} = \{\boldsymbol{w} \mid \boldsymbol{w} \in [0,1]^k, \sum_{i=1}^{k} w_i = 1\}, \ |W| = n, \tag{5}$$

The cost C_{rj} of assigning the individual $\boldsymbol{x}_j$ to the weight vector $\boldsymbol{w}_r$ is given by:

$$C_{rj} = \max_{i=1,\dots,k} w_{ri} \times \tilde{f}_i(\boldsymbol{x}_j), \ r = 1, ..., n, \ j = 1, ..., 2n. \tag{6}$$

Algorithm 1: Hungarian Differential Evolution (HDE)

```
   Input  : MOP, population size ($n$), maximum number of generations ($g_{max}$),
            parameters $C_r$ and $F$ for DE/$rand$/1/$bin$
   Output: $P_{g_{max}}$ (approximation of the $\mathcal{P}^*$ and $\mathcal{PF}^*$)
1  Generate initial population $P_1$ randomly;
2  Evaluate each individual in $P_1$;
3  $W \leftarrow$ Generate $n$ weight vectors using Algorithm 2;
4  for $g = 1$ to $g_{max}$ do
5      $P_g^* \leftarrow$ Generate offspring using $P_g$ and DE/$rand$/1/$bin$;
6      Evaluate each individual in $P_g^*$;
7      $Q_g \leftarrow P_g \cup P_g^*$;
8      Calculate $\boldsymbol{z}^{max}$ and $\boldsymbol{z}^{min}$ by (3) Normalize objectives of each individual in
       $Q_g$ by (4);
9      Generate the cost matrix $C$ by (6) using $Q_g$ and $W$;
10     $\mathcal{I} \leftarrow$ Obtain the best assignment in $C$ using the Hungarian Method;
11     $P_{g+1} \leftarrow \{\boldsymbol{x}_i \mid i \in \mathcal{I}, \boldsymbol{x}_i \in Q_g\}$;
12 end
```

The matrix C indicates how each individual is suitable to represent each region of the Pareto Front approximation. The solution to our assignment problem is found by identifying the combination of values in C resulting in the smallest sum, subject to certain constraints. These conditions are:

1. Exactly one value must be chosen in each row; this ensures that only one individual is assigned to each position on the Pareto Front.
2. At most one value can be selected in each column; this ensures that no individual is assigned to more than one position.

The matrix C and the two above constraints are formally represented by (2) as a linear programming problem. The solution to this problem is obtained by the extended Kuhn-Munkres algorithm for rectangular matrices, presented in Section 2. The matrix that solves (2) represents the individuals assigned to each weight vector such that it minimizes the total cost of the assignment, allowing to retain the best n individuals to approximate the Pareto Front. The pseudo-code of our proposed approach is depicted in Algorithm 1.

3.1 Generation of weight vectors using Uniform Design

There exist several MOEAs [23, 18, 8] that require a set of weight vectors uniformly scattered on the $(k-1)$-unit simplex to obtain solutions along the entire Pareto Front in a k-objective optimization problem. A variety of methods to obtain an evenly distributed subset of weights in a simplex are available in the specialized literature [10]. The simplex-lattice design method [20] is the approach that has been the most commonly adopted in MOEAs. However, at least three problems can be identified in this method [10]. First, the weight vectors are not very uniformly distributed. Second, there are too many vectors at the boundary of the domain. Furthermore, the number of vectors generated increases nonlinearly with the number of objectives. That is, if H divisions are considered along each objective, the total number of weight vectors (hence the population size) in a k-objective problem is given by: $\binom{H+k-1}{k-1}$. Due to this, some MOEAs have used other methods to generate an arbitrary number of weight vectors well-distributed over a simplex. In [18] a hypervolume-based weight vector generation is proposed. This method produces well-distributed vectors maximizing the hypervolume covered by them in objective space. A different idea was proposed in [21], where the uniform design (UD) [10] and good lattice point (*glp*) [14] methods are combined to set the weight vectors. Nevertheless, both the hypervolume and the *glp* method have a high computational cost when the number of objectives grows.

Uniform design is a space filling design method that seeks experimental points to be uniformly scattered on the domain [10]. In uniform design, a set of points is considered uniformly spread throughout the entire domain if it has a small discrepancy, where discrepancy is a numerical measure of scatter. Fang and Wang [10] presented different methods for generating points that can be applied to the generation of a set of space-filling design points. Among them, we have the good lattice point (*glp*) method and Hammersley method [12], both of which are efficient quasi Monte-Carlo methods.

We propose to generate weight vectors using uniform design combined with Hammersley method. This algorithm allows a more uniform distribution of the weight vectors over the space than the simplex-lattice method, and the population size neither increases nonlinearly with the number of objectives nor considers a formulaic setting. Additionally, Hammersley method provides a set of design points with low discrepancy similar to the *glp* method, but at a much lower computational cost [10].

The Hammersley method is based on the p-adic representation of natural numbers: Any positive integer m can be uniquely expressed using a prime base $p \geq 2$ as

$$m = \sum_{i=0}^{r} b_i \times p^i, \quad 0 \leq b_i \leq p-1, \quad i = 0, \dots, r, \tag{7}$$

where $p^r \leq m < p^{r+1}$. Then, for any integer $m \geq 1$ with representation (7), let

$$y_p(m) = \sum_{i=0}^{r} b_i \times p^{-(i+1)}, \tag{8}$$

where $y_p(m) \in (0,1)$ and is known as the radical inverse of m base p. Let $k \geq 2$ and $p_1, \dots, p_{k-1}$ be $k-1$ distinct prime numbers, the Hammersley set consisting of n points uniformly scattered on $[0,1]^k$ is given by

$$\boldsymbol{x}_i = \left[\frac{2i-1}{2n}, y_{p_1}(i), \dots, y_{p_{k-1}}(i) \right]^T, \; i = 1, \dots, n. \tag{9}$$

In [22], it was proposed to use uniform design for experiments with mixture (UDEM) that seek points to be uniformly scattered in the domain $\mathbb{W}$

Algorithm 2: Generation of weight vectors

```
Input  : number of objectives ($k$), number of weights ($n$)
Output: $W$ (set of weight vectors with low-discrepancy)
1  $p \leftarrow$ array with the first $k-2$ prime numbers;
2  $U \leftarrow \emptyset$;
3  for $i = 1$ to $n$ do
4      $u_{i1} \leftarrow (2i-1)/2n$;
5      for $j = 2$ to $k-1$ do
6          $u_{ij} \leftarrow 0$;
7          $f \leftarrow 1/p_{j-1}$;
8          $d \leftarrow i$;
9          while $d > 0$ do
10             $u_{ij} \leftarrow u_{ij} + f \times (d \mod p_{j-1})$;
11             $d \leftarrow \lfloor d/p_{j-1} \rfloor$;
12             $f \leftarrow f/p_{j-1}$;
13         end
14     end
15     $U \leftarrow U \cup \{u\}$;
16 end
17 $W \leftarrow$ Apply the transformation (10) to $U$;
```

defined by (5). They employed the transformation method for the construction of such uniform design. This method requires a set of vectors $U = \{\boldsymbol{u}_i = [u_{i1}, ..., u_{i(k-1)}]^T, i = 1, ..., n\} \subset [0,1]^{k-1}$ with small discrepancy. In our proposal, the Hammersley method is used to obtain U and then to apply the transformation

$$\begin{aligned} w_{ti} &= (1 - u_{ti}^{\frac{1}{k-i}}) \prod_{j=1}^{i-1} u_{tj}^{\frac{1}{k-j}}, \quad i = 1, ..., k-1, \\ w_{tk} &= \prod_{j=1}^{k-1} u_{tj}^{\frac{1}{k-j}}, \quad t = 1, ..., n. \end{aligned} \tag{10}$$

Then $\{\boldsymbol{w}_t = [w_{ti}, ..., w_{tk}]^T,\ t = 1, ..., n\}$ is a uniform design on $\mathbb{W}$. The pseudocode of the algorithm used to generate weight vectors is presented in Algorithm 2.

4 Experimental Results

We validated our proposed HDE comparing its performance with respect to two MOEAs representative of the state-of-the-art in the area: the multi-objective evolutionary algorithm based on decomposition [23] (MOEA/D) and the S metric selection Evolutionary Multi-Objective Algorithm [2] (SMS-EMOA). Since the SMS-EMOA requires a considerably large amount of computational time in problems with more than five objectives, we also include in this comparative study a version of this MOEA (called appSMS-EMOA) that uses the algorithm proposed in [1] to approximate the hypervolume contributions using Monte Carlo sampling.

In our experiments, we adopted 12 test problems, consisting of five bi-objective problems taken from the Zitzler-Deb-Thiele (ZDT) test suite [25] and seven test problems having from two to ten objective functions taken from the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [9]. In the problems ZDT1-3 the number of decision variables is 30; ZDT4 and ZDT6 have 10 variables. In the DTLZ test problems, the total number of variables is given by $n = m + k - 1$, where $m = 2, ..., 10$ is the number of objectives and k was set to 10 for DTLZ1-6 and 20 for DTLZ7.

In order to assess the performance of each MOEA, we selected the hypervolume indicator as a performance measure. The hypervolume is the size of the space covered by the Pareto optimal solutions, thus capturing both convergence and diversity in a single value [24]. The hypervolume can differentiate between degrees of complete outperformance of two sets [5]. To calculate the hypervolume indicator, we used the reference points $\boldsymbol{y}_{ref} = [y_1, \cdots, y_m]$ such that: $y_i = 1.1$ for all the ZDT problems, and for DTLZ1, DTLZ2 and DTLZ4; $y_i = 3$ for DTLZ3, DTLZ5 and DTLZ6; and $y_i = 7$ for DTLZ7. We also considered the running time of each algorithm. Running times as a measure of computational cost are particularly relevant when increasing the number of objectives. In order to achieve more confident results, each MOEA was executed 30 times for each problem instance, and we report here their average hypervolume values and their average running times.

Our proposed HDE uses the variation operators of differential evolution and, therefore, it uses its same parameters. The parameters adopted in our experiments were: $F = 1.0$ and $Cr = 0.4$. The recombination operators of MOEA/D, SMS-EMOA and appSMS-EMOA are simulated binary crossover and polynomial-based mutation. Their corresponding parameters were set as follows: crossover probability $p_c = 1.0$, mutation probability $p_m = 1/n$, where n is the number of decision variables; the distribution indexes were set as: $\eta_c = 20$ and $\eta_m = 20$. MOEA/D used the Tchebycheff approach with a neighborhood size of 20. The number of samples for the Monte Carlo estimation in appSMS-EMOA was set to 10^4. The algorithms HDE, SMS-EMOA and appSMS-EMOA can use an arbitrary population size, but in MOEA/D the population size increases nonlinearly with the number of objectives. For this reason, we used different population sizes. In the ZDT bi-objective problems the population size was set to 100. For the DTLZ problems with 2, 3, 4 and 8 objectives, the population size was set to 120. For problems having 5 and 6 objectives, the population size was set to 126. Finally, for problems having 7, 9 and 10 objectives, the population size was set to 210, 165 and 220, respectively. The maximum number of generations adopted in the ZDT test problems was 200, and we used 300 for the DTLZ test problems. It is important to note that SMS-EMOA was not applied to problems with more than 5 objectives due to its high computational cost.

4.1 Discussion of Results

First, we will review our results in the ZDT test problems. Table 1 provides the average hypervolume of each compared MOEA for each test problem. The best results are presented in **boldface**. From these results, we can see that our proposed HDE outperformed all the other MOEAs in all the test problems, except for ZDT3, where SMS-EMOA achieved a slightly higher hypervolume value.

Table 1: Results obtained in the ZDT test problems. We show the average hypervolume values obtained over 30 independent runs.

Problem	HDE	MOEA/D	appSMS-EMOA	SMS-EMOA
ZDT1	**0.871748**	0.863668	0.868213	0.871514
ZDT2	**0.538383**	0.517640	0.528167	0.537282
ZDT3	1.327721	1.298709	1.296754	**1.328633**
ZDT4	**0.833392**	0.602010	0.804054	0.822489
ZDT6	**0.504490**	0.496180	0.487527	0.493889

In Table 2 we present the average hypervolume for the DTLZ test problems. Figure 4 shows the average runtime for each instance of the DTLZ problems having from two to ten objective functions. In the DTLZ1 and DTLZ3 problems, HDE outperforms the other MOEAs for every number of objective functions. The search space in these two problems contains $(11^k - 1)$ and $(3^k - 1)$ local Pareto fronts, respectively ($k = 10$ in our experiments). This makes difficult to converge to the true Pareto front. SMS-EMOA and appSMS-EMOA are unable to converge to the true Pareto front in any instance of the DTLZ3 problem. Additionally, appSMS-EMOA does not perform well in DTLZ1.

For DTLZ2 and DTLZ4, SMS-EMOA performs better than the other MOEAs in instances having from two to five objectives, but its runtime is of up to 20 hours in DTLZ2 with five objectives and it reaches up to four days in DTLZ4. appSMS-EMOA obtains the best results in the instances with six and seven objectives, but requires several minutes per run. In DTLZ2 and DTLZ4 with more than seven objectives, HDE outperforms all the other algorithms and requires only seconds per run. A similar observation can be made for the problem DTLZ5, where SMS-EMOA obtains the best results in the instances having from two to five objectives, whereas for more than five objectives HDE performs better than the other MOEAs except for eight objectives.

The main feature of DTLZ5 and DTLZ6 is that the Pareto front is a curve (it loses dimensionality). However, DTLZ6 is considered to be harder to solve than DTLZ5, because MOEAs tend to have more difficulties to reach the true Pareto front with this problem. In DTLZ6, HDE outperforms the other MOEAs for all the instances having from two to ten objectives, and appSMS-EMOA presented a poor performance. HDE also obtained the best results in DTLZ7, which has a disconnected Pareto front. For all instances with three objectives or more, HDE outperformed the other algorithms; only for two objectives SMS-EMOA achieved a slightly higher hypervolume value.

Table 2: Results obtained in the DTLZ test problems. We show the average hypervolume values obtained over 30 independent runs.

N^o Obj.	HDE	MOEA/D	appSMS-EMOA	SMS-EMOA
		DTLZ1		
2	**1.0833e+0**	1.0662e+0	1.0286e+0	1.0487e+0
3	**1.3022e+0**	1.2650e+0	8.6233e–1	1.1704e+0
4	**1.4565e+0**	1.2713e+0	2.9529e–2	1.4536e+0
5	**1.6084e+0**	1.3297e+0	0.0000e+0	1.6041e+0
6	**1.7605e+0**	1.5175e+0	1.1103e–4	-
7	**1.9466e+0**	1.9416e+0	0.0000e+0	-
8	**2.1435e+0**	1.9369e+0	0.0000e+0	-
9	**2.3579e+0**	2.2682e+0	0.0000e+0	-
10	**2.5937e+0**	2.5592e+0	0.0000e+0	-
		DTLZ2		
2	4.2060e–1	4.2087e–1	4.2013e–1	**4.2161e–1**
3	7.3603e–1	7.1504e–1	7.4889e–1	**7.6251e–1**
4	9.8341e–1	8.8689e–1	1.0195e+0	**1.0526e+0**
5	1.2229e+0	1.1406e+0	1.2570e+0	**1.3090e+0**
6	1.4462e+0	1.2123e+0	**1.4700e+0**	-
7	1.7219e+0	1.2972e+0	**1.7339e+0**	-
8	**1.9028e+0**	1.2018e+0	1.8572e+0	-
9	**2.1706e+0**	1.3226e+0	2.1051e+0	-
10	**2.4603e+0**	1.4329e+0	2.3859e+0	-
		DTLZ3		
2	**8.2085e+0**	8.1148e+0	0.0000e+0	0.0000e+0
3	**2.6404e+1**	2.6067e+1	0.0000e+0	0.0000e+0
4	**8.0515e+1**	7.6359e+1	0.0000e+0	0.0000e+0
5	**2.4260e+2**	2.2909e+2	0.0000e+0	3.9658e–1
6	**7.2861e+2**	6.8984e+2	0.0000e+0	-
7	**2.1854e+3**	2.1643e+3	0.0000e+0	-
8	**6.5606e+3**	6.2456e+3	0.0000e+0	-
9	**1.9683e+4**	1.9186e+4	0.0000e+0	-
10	**5.9049e+4**	5.7783e+4	0.0000e+0	-
		DTLZ4		
2	4.2050e–1	4.2087e–1	4.2026e–1	**4.2161e–1**
3	7.3349e–1	7.1758e–1	7.4999e–1	**7.6254e–1**
4	9.8347e–1	8.8985e–1	1.0249e+0	**1.0527e+0**
5	1.2290e+0	1.1440e+0	1.2660e+0	**1.3094e+0**
6	1.4518e+0	1.3216e+0	**1.4880e+0**	-
7	1.7290e+0	1.4835e+0	**1.7467e+0**	-
8	**1.9065e+0**	1.3559e+0	1.8973e+0	-
9	**2.1773e+0**	1.4883e+0	2.1206e+0	-
10	**2.4646e+0**	1.5820e+0	2.4016e+0	-
		DTLZ5		
2	8.2106e+0	8.2108e+0	8.2102e+0	**8.2116e+0**
3	2.3979e+1	2.3967e+1	2.3985e+1	**2.3990e+1**
4	7.1549e+1	7.1247e+1	7.1497e+1	**7.1856e+1**
5	2.1419e+2	2.0875e+2	2.1385e+2	**2.1567e+2**
6	**6.4008e+2**	6.1645e+2	6.3956e+2	-
7	**1.9271e+3**	1.8336e+3	1.9188e+3	-
8	5.6978e+3	5.4432e+3	**5.7225e+3**	-
9	**1.7197e+4**	1.6307e+4	1.7171e+4	-
10	**5.1725e+4**	4.8723e+4	5.1545e+4	-
		DTLZ6		
2	**8.2108e+0**	8.0197e+0	1.1719e+0	3.1194e+0
3	**2.3982e+1**	2.3487e+1	2.1721e+1	2.3745e+1
4	**7.1345e+1**	6.9232e+1	3.7674e+1	6.7598e+1
5	**2.1324e+2**	1.9631e+2	3.2082e+1	1.9830e+2
6	**6.3859e+2**	5.7393e+2	4.7076e+0	-
7	**1.9204e+3**	1.7051e+3	3.5091e–1	-
8	**5.6590e+3**	4.9692e+3	1.0260e+0	-
9	**1.7070e+4**	1.5059e+4	0.0000e+0	-
10	**5.1444e+4**	4.4826e+4	0.0000e+0	-

Continued on next page

Table 2: – Continued from previous page

N^o Obj.	HDE	MOEA/D	appSMS-EMOA	SMS-EMOA
		DTLZ7		
2	3.1881e+1	3.0554e+1	3.1881e+1	**3.1884e+1**
3	**2.0053e+2**	1.8413e+2	1.9708e+2	1.9931e+2
4	**1.2267e+3**	8.8240e+2	1.1081e+3	1.1598e+3
5	**7.2516e+3**	3.9693e+3	4.5253e+3	6.2818e+3
6	**3.9558e+4**	1.7243e+4	1.4071e+4	-
7	**2.0094e+5**	5.0926e+4	5.2148e+4	-
8	**7.2712e+5**	8.4243e+4	4.6362e+5	-
9	**3.1360e+6**	2.8190e+5	1.6500e+6	-
10	**9.1999e+6**	1.2141e+6	4.5446e+6	-

5 Conclusions and Future Work

We have proposed a novel selection scheme for MOEAs. Our approach transforms the selection mechanism of a MOEA into an assignment problem using a set of well-distributed points on a unit simplex. The obtained assignment problem is solved with the Kuhn-Munkres algorithm. We have also suggested an algorithm based on uniform design to generate a set of weight vectors more uniformly scattered than those obtained by the simplex-lattice method. Our experimental results indicate that our proposed HDE outperforms MOEA/D in several test problems, and is competitive (outperforming it in several instances) with respect to SMS-EMOA, while requiring a significantly lower computational time.

As part of our future work, we intend to study other (computationally inexpensive) uniform design methods to generate a set of points more uniformly distributed. We also plan to analyze other methods for solving assignment problems at a lower computational cost.

References

1. J. Bader and E. Zitzler. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, Spring, 2011.
2. N. Beume, B. Naujoks, and M. Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, September 2007.
3. F. Bourgeois and J.-C. Lassalle. An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices. *Commun. ACM*, 14(12):802–804, December 1971.
4. K. Bringmann and T. Friedrich. Approximating the least hypervolume contributor: NP-hard in general, but fast in practice. *Theoretical Computer Science*, 425:104–116, March 2012.
5. D. Brockhoff, T. Friedrich, and F. Neumann. Analyzing Hypervolume Indicator Based Algorithms. In G. Rudolph, T. Jansen, S. Lucas, C. Poloni, and N. Beume, editors, *Parallel Problem Solving from Nature PPSN X*, volume 5199 of *Lecture Notes in Computer Science*, pages 651–660. Springer Berlin Heidelberg, September 2008.

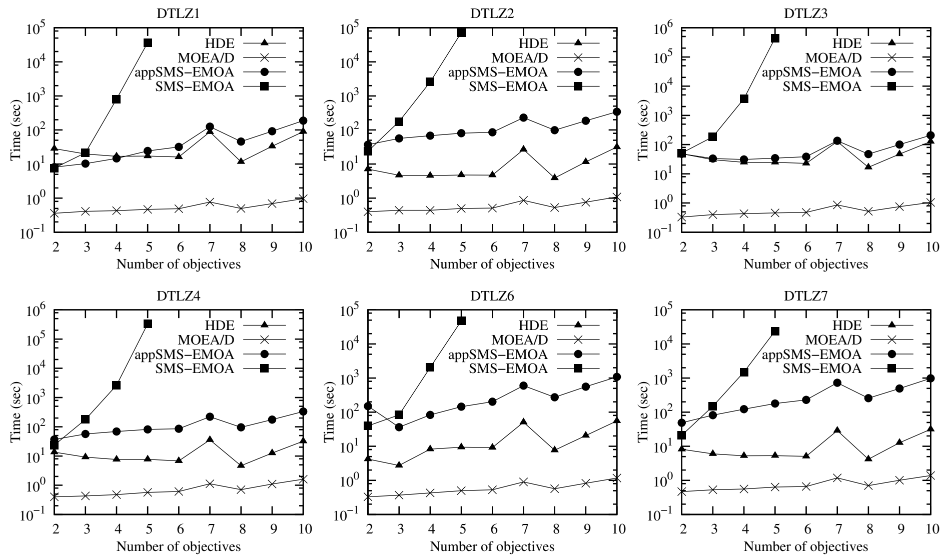

Fig. 4: Average runtime over 30 independent runs of HDE, MOEA/D, SMS-EMOA and appSMS-EMOA in the DTLZ test suite.

6. D. Brockhoff, T. Wagner, and H. Trautmann. On the Properties of the R2 Indicator. In *2012 Genetic and Evolutionary Computation Conference (GECCO'2012)*, pages 465–472, Philadelphia, USA, July 2012. ACM Press.
7. C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, Sept. 2007.
8. 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. *IEEE Transactions on Evolutionary Computation*, 18(4):577–601, August 2014.
9. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145. Springer Berlin Heidelberg, USA, 2005.
10. K. T. Fang and Y. Wang. *Number-Theoretic Methods in Statistics*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1994.
11. M. Fleischer. The Measure of Pareto Optima. Applications to Multi-objective Metaheuristics. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization (EMO 2003)*, volume 2632 of *Lecture Notes in Computer Science*, pages 519–533. Springer Berlin Heidelberg, Faro, Portugal, April 2003.
12. J. M. Hammersley. Monte-Carlo methods for solving multivariable problems. *Annals of the New York Academy of Sciences*, 86(3):844–874, 1960.
13. H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *2008 IEEE Congress on Evolutionary Computation*

CEC'2008 (IEEE World Congress on Computational Intelligence), pages 2424–2431, Hong Kong, June 2008.

14. N. M. Korobov. The approximate computation of multiple integrals. *Doklady Akademii Nauk SSSR*, 124:1207–1210, 1959.
15. H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, Mar. 1955.
16. H. Li and Q. Zhang. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302, April 2009.
17. J. Munkres. Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, Mar. 1957.
18. D. H. Phan and J. Suzuki. R2-IBEA: R2 Indicator Based Evolutionary Algorithm for Multiobjective Optimization. In *IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 1836–1845, 2013.
19. K. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization*. Natural Computing Series. Springer-Verlag, Secaucus, NJ, USA, 2005.
20. H. Scheffé. Experiments with mixtures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(2):344–360, 1958.
21. Y.-Y. Tan, Y.-C. Jiao, H. Li, and X.-K. Wang. MOEA/D + uniform design: A new version of MOEA/D for optimization problems with many objectives. *Computers & Operations Research*, 40(6):1648–1660, June 2013.
22. Y. Wang and K. T. Fang. Number-Theoretic Method in Applied statistics (II). *Chinese Annals of Mathematics. Serie B*, 11:859–914, 1990.
23. Q. Zhang and H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, December 2007.
24. E. Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.
25. E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.
26. E. Zitzler and S. Künzli. Indicator-based Selection in Multiobjective Search. In X. Yao, E. Burke, J. Lozano, J. Smith, J. Merelo-Guervs, J. Bullinaria, J. Rowe, P. Tio, A. Kabn, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature PPSN VIII*, volume 3242 of *Lecture Notes in Computer Science*, pages 832–842. Springer Berlin Heidelberg, Birmingham, UK, September 2004.