GeDEA-II: A Simplex Crossover Based Evolutionary Algorithm Including the Genetic Diversity as Objective

Claudio Comis Da Ronco and Ernesto Benini

Abstract—**The key issue for an efficient and reliable multi-objective evolutionary algorithm is the ability to converge to the True Pareto Front with the least number of objective function evaluations, while covering it as much as possible. To this purpose, in a previous paper performance comparisons showed that the Genetic Diversity Evolutionary Algorithm (GeDEA) was at the same level of the best state-of-the-art MOEAs due to it intrinsic ability to properly conjugate exploitation of current non-dominated solutions and the exploration of the search space. In this paper, an improved version, namely the GeDEA-II, is proposed which features a novel crossover operator, the Simplex-Crossover, and a novel mutation operator, the Shrink-Mutation.**

GeDEM operator was left unchanged and completed using the non-dominated-sorting based on crowding distance. The comparison among GeDEA-II and GeDEA, as well as with three other modern elitist methods, on different extremely multidimensional test problems, clearly indicates that the performance of GeDEA-II is, at least in these cases, superior. In addition, authors aimed at putting in evidence the very good performance of GeDEA-II even in extremely multidimensional landscapes. To do this, four test problems were considered, and the GeDEA-II performance tested as the number of decision variables was increased. In particular, ZDT test functions featured a number of decision variables ranging from the original proposed number up to 1000, whereas on $DTLZ$ the decision variables were increased up to 100 times the original proposed number. Results obtained contribute to demonstrate further the GeDEA-II breakthrough performance.

Index Terms—**Evolutionary algorithms, Simplex Crossover, Shrink Mutation, Pareto optimality, multi objective optimization, Empirical - Comparison.**

I. Introduction

In the past, several of MOEAs were proposed, e.g., Multi Objective Genetic Algorithm (MOGA) [1], Niched Pareto Genetic Algorithm (NPGA) [2] and Non-dominated Sorting Genetic Algorithm (NSGA) [3], which demonstrated the capability of evolutionary multi-objective optimization (EMO) algorithms to approximate the set of optimal trade-offs in a single optimization run. These approaches did not incorporate elitism explicitly, but a few years later the importance of this concept in multi-objective search was recognized and supported experimentally [4]. A couple of elitist MOEAs, which soon became state-of-the-art, were Strength Pareto Evolutionary Algorithm (SPEA) [5], [6] and Pareto Archived Evolution Strategy (PAES) [7]. SPEA, an acronym for Strength Pareto Evolutionary Algorithm, was among the first techniques that were extensively compared to several existing evolution-based methods [6], [4]. Later on, further progress has been made and the new proposed methods, featuring different diversity preservation mechanisms, for instance NSGA-II [8], PESA [9] and SPEA2 [10] were shown to outperform SPEA and NSGA on certain test problems. GeDEA [11] algorithm, strictly designed around the genetic diversity preservation mechanism called GeDEM, proved to be able to compete and, in some cases, to outperform, the aforementioned EAs as far as speed of convergence and covering uniformity of the Pareto Front are concerned. In fact, the common drawback of all of the previously mentioned MOEAs is the huge amount of objective function evaluations (or number of generations) required to reach and sufficiently cover the Pareto Front.

To try to overcome this common weakness, during the last decade several authors started hybridizing evolutionary algorithms (EAs) with local search (LS) operators, giving rise to the so-called *Memetic Algorithms* (MAs), (see [12] for a review, [13] for a collection of recent algorithmic and theoretical work, and [14] for a comprehensive bibliography).

In [15], authors proposed a hybridized version of NSGA-II, coupled with a classical Sequential Quadratic Programming (SQP) procedure to achieve better performance. As clearly claimed by the authors in [16], "the main drawback of this approach is that SQP requires calculating the function gradient and the optimum step length at every iteration, an operation that can be costly for a practical engineering problem". A valid alternative to the time-consuming calculation of gradient information is constituted by the direct local search method. Among the direct search methods, the Simplex method proposed for the first time by the authors in [17] and subsequently improved by the authors in [18], is one of the most popular methods due to its simplicity and ease of encoding.

In [19], the authors worked out an hybrid method, called continuous hybrid algorithm, performing the exploration with a Genetic Algorithm (GA), and the exploitation with a Nelder-Mead Simplex algorithm.

In [20], the authors integrated Nelder-Mead simplex search method [18] with genetic algorithm in order to combine the local search capabilities of the former, and the exploratory behavior of the latter.

Moreover, several works were presented, with the purpose to extend these concepts to multi-objective problems.

Authors in

Recently, in [21] an hybrid Simplex MOEA has been proposed, which uses three subsets to evolve simultaneously. The first two subsets are constituted by individuals calculated

C. Comis Da Ronco is with HIT09 S.r.l, Galleria Storione 8, 35131 Padova, ITALY e-mail: c.comis@hit09.com

Ernesto Benini is with the Department of Industrial Engineering, University of Padova, Via Venezia 1, 35131, Padova, ITALY e-mail: ernesto.benini@unipd.it

via simplex-based local search method to achieve faster convergence and better diversity, whereas the third one gathers together individuals generated by means of ordinary genetic operators to avoid premature convergence.

In [16], authors proposed their version of hybrid MOEA, which starts with a randomly generated population with a user-defined size. Using this initial population, a few generations of NSGA-II are carried on. The local search is activated only after all the individuals of the current population are located at the first non-domination front. When activated, the local search operates only on a subset of the current population. The selected solutions in the current population are then replaced by the improved solutions found by the local search, creating a locally improved population. The locally improved population is used then as an initial population for the next few generations by NSGA-II. Once again, the individuals created by means of the local search and those ones created with the variation operators of the MOGA are created at two different moments, and merged together into the final population.

In spite of the different frameworks, in all the previously mentioned works, the local search, based on the Simplex algorithm, and the global exploration based on the EA, are performed separately, in a sequential manner, that is, a point of the search space is calculated via either the first or the latter.

In the authors' opinion, the previously mentioned examples of hybridization with local search often degrade the global search ability of MOEAs. Moreover, local search based on the Nelder and Mead requires additional and several functions evaluations.

In this paper, GeDEA-II is presented, aiming at reducing the potential weaknesses of its predecessor and competitors, while retaining its superior performance, that is, a good balance between exploration and exploitation. In this work, a different approach is proposed to combine the EA-based global search and the Simplex theory, since global exploration and local search are intimately related and performed simultaneously, in such a way they take advantage from each other. In details, the individuals created by the proposed algorithm via the Simplex-based crossover, undergo mutation in a subsequently step, so as to promote global search capabilities of the algorithm. Moreover, important modifications have been brought about to the original Simplex theory, in order to enhance further the local search capabilities without penalizing the exploration of the search space.

The main differences of GeDEA-II in comparison with GeDEA regard its new Tournament-Selection operator, its new Simplex-Crossover operator, and its new Shrink-Mutation operator. The diversity preserving mechanism, the Genetic Diversity Evaluation Method (GeDEM) already used in the GeDEA release, was retained in GeDEA-II and left unchanged due to its superior performance.

The paper is structured as follows. Section II presents a brief description of the main characteristics of the competitors MOEAs. In Section III, the main characteristics of the GeDEA are presented to prepare the ground for the GeDEA-II, whose framework is introduced and described in detail in Section IV. Finally, in Section V a systematic comparison between GeDEA-II and other state-of-the-art MOEAs is presented, following the guidelines proposed in [4], and then we describe our experimental results.

II. Multiobjective Evolutionary Algorithms and the Problem of Diversity Preservation

Nowadays, two features are demanded of an efficient and robust MOEA, which are:

1) To perform the optimization reducing at a minimum the overall optimization time;
2) To perform multiple runs while achieving the same results.

The first feature is important when considering a MOEA actual application to a real-world engineering problem, where overall computational time is significant. The second feature is referred to as *repeatability*, and it is important in order to judge the efficiency of the EA under investigation. As discussed in Section V-D, GeDEA-II proves to have both these characteristics.

In the following, the authors briefly analyze the constitutive framework of the respective MOEAs, along with the strategies implemented to promote diversity. For comprehensive overviews of evolutionary approaches to multi-objective optimization the reader is referred to the following more specific studies [22], [23].

Many EAs for multi-objective optimization have been proposed [4], [22], [9]. Probably, the most popular MOEAs today are the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [10] and the Non-dominated Sorting Genetic Algorithm-II [8], which have been used for comparison in the Experimental results Section. As clearly stated in [24], the two algorithms similarly maintain a separate population of size N (current population, or offspring population) and a fixed-capacity archive (previous population, or parent population), often (as in NSGA-II) also dimensioned *N*. In each generation, the current and the previous populations are merged together, and undergo the process of elite preservation. The population constituting the new generation is filled by taking the best-ranked solutions from the merged list. Rank conflicts are resolved via a diversity metric. Individuals are also subject to tournament selection, crossover, and mutation to form the population for the next generation. The main difference between the two is the way elite preservation is applied. NSGA-II invokes a procedure called non-dominated sorting. Non-dominated sorting assigns domination ranks to each individual solution in a population, in such a manner that solutions are assigned lower ranks than the ones they dominate. The non-dominated sorting procedure incorporates a diversity preservation mechanism, introduced in Section IV-C, which estimates the density of solutions in the objective space, and the crowded comparison operator, which guides the selection process towards a uniformly spread Pareto frontier.

SPEA2 on the other hand, as clearly stated in [5], incorporates a fine-grained fitness assignment strategy which takes into account for each individual the number of individuals that dominate it and the number of individuals by which it is dominated. It then uses a nearest neighbor density estimation technique which guides the search more efficiently, and finally performs an enhanced archive truncation that guarantees the preservation of boundary solutions.

Another interesting multi-objective evolutionary algorithm is the Indicator-Based Evolutionary Algorithm (IBEA), proposed in [25]. It is a MOEA that calculates fitness values by comparing individuals on the basis of a quality indicator. Thereby, no particular diversity preservation techniques such as fitness sharing, clustering, etc. is necessary.

III. Genetic Diversity Evolutionary Algorithm (GeDEA)

As GeDEA (Genetic Diversity Evolutionary Algorithm) forms the basis for GeDEA-II, we give a brief summary of the algorithm here. For a more detailed description, the interested reader is referred to [11]. The Genetic Diversity Evolutionary Algorithm (GeDEA) is a framework that is strictly designed around GeDEM to exalt its characteristics. Some of the design choices follow from the basic features of GeDEM (e.g., the replacement of clones, the use of an elitist strategy), the others are inspired by the will to make things as simple as possible, and neither introducing arbitrary parameters nor using sophisticated heuristics. To briefly introduce the GeDEM principle, it is worth to conceptually go back to the beginning of Section 2, where it was explained that the multi-objective optimization process has two objectives, which are themselves conflicting: the convergence to the Pareto-optimal set and the maintenance of genetic diversity within the population. The basic idea of GeDEM is to actually use these objectives during the evaluation phase and to rank the solutions with respect to them, emphasizing the non-dominated solutions as well as the most genetically different.

When the GeDEM is applied, the actual ranks of the solutions are determined maximizing (i) the ranks scored with respect to the objectives of the original MOOP, the non-dominated solutions having the highest rank, and (ii) the values assigned to each individual as a measure of its genetic diversity, calculated according to the chosen distance metric, i.e. the (normalized) Euclidean distance in the decision variable space. The structure of GeDEA follows the main steps of a $(\mu + \lambda)$ Evolution Strategy [26]. The evolution, however, is considered at its genotypic level, with the traditional binary coding of the decision variables. In the following the framework of the GeDEA is recalled for clarity.

- **Step 1**: An initial population of μ individuals is generated at random.
- **Step 2**: A mating pool of 2λ individuals is formed, each individual having the same probability of being selected.
- **Step 3**: λ offspring are generated by crossover. Some bits of the offspring are also randomly mutated with a probability p_{mut}.
- **Step 4**: The whole population of $\mu + \lambda$ individuals is checked to discover possible clones. These clones are removed and replaced with new randomly generated individuals (this is done to encourage the exploration of the search space and also to have the algorithm evaluate, for convenience, new λ different offspring every generation; still the occurrence of clones birth is not so frequent if clones are removed generation after generation). This task is accomplished every generation, just before the objective functions evaluation, in order to prevent the same individual is evaluated more than once.
- **Step 5**: The objective function values of the $\mu + \lambda$ individuals are evaluated and the non-dominated sorting procedure presented in [27] is performed to assign the ranks to the solutions according to the objectives of the MOOP.
- **Step 6**: The whole population of $\mu + \lambda$ individuals is processed to determine the value of the distance-based genetic diversity measure for each individual.
- **Step 7**: GeDEM is applied according to the ranks scored in Step 5 and the values of the diversity measure assigned in Step 6. The non-dominated sorting procedure presented in [27] is used again to assign the ranks.
- **Step 8**: The best μ solutions among parents and offspring, according to the ranks assigned in Step 7 by GeDEM, are selected for survival and the remaining λ are eliminated.
- **Step 9**: If the maximum number of generations is reached then stop, else go to Step 2.

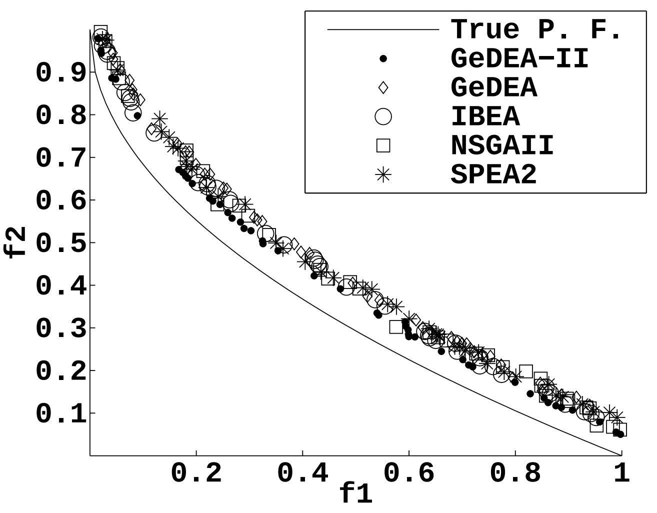


Fig. 1. Approximate Pareto-optimal set reached by GeDEA-II, GeDEA, IBEA, NSGA-II and SPEA2 provided with the same evolutionary operators, on ZDT_1 test function.

While GeDEA-II shares with its predecessor the same framework, it is presented in this work in a real-coded fashion. Hence, it features the same parameters representation of the competitor algorithms already presented in Section II. Moreover, this choice was made in view of using it for solving real-world engineering optimization problems. In order to prove the efficiency of the $(\mu + \lambda)$ Evolution Strategy, whose steps are strictly followed in GeDEA-II, a comparison with the competitor algorithms framework was performed. For each algorithm, the same crossover, mutation and selection operators were exploited, that is the SBX Crossover [28], the Polynomial Mutation (implemented as described in [29]) and the Tournament Selection [30] operators, respectively. Results of these comparisons are depicted in Fig. 1. The results presented here refer to the ZDT_1 bi-objective test problem, which involves 30 decision variables, and is thoroughly described in V-A. Results hint that the GeDEA-II framework provides remarkable results in terms of both convergence and Pareto set coverage, and therefore underlies the good performance of the algorithm itself.

IV. Genetic Diversity Evolutionary Algorithm-II (GeDEA-II)

GeDEA proved to be an efficient algorithm, able to explore widely the search space, while exploiting the relationships among the solutions. In order to enhance GeDEA algorithm performance further, several main features were added to the previous GeDEA version, yet retaining its constitutive framework. The main innovation is the novel crossover function, namely the Simplex-crossover, which takes place in lieu the previous Uniform crossover. Novel selection and mutation operators were also developed. The first one, namely the Tournament-selection operator, allows exploring the design space more effectively. The second one, namely the Shrink-mutation, allows exploring more effectively the design space. The remaining steps characterizing GeDEA algorithm, in particular the GeDEM, were left unchanged.

The latter was integrated with the Non-Dominating sorting procedure based on the crowding distance. The following sections present a detailed overview of the work already described in [31].

A. The SIMPLEX Crossover

In many EAs, a recombination with two parents is commonly used to produce offspring. At the end of the 90's, in several studies the use of more than two parents for recombination in EAs have been reported [32], [33], [34].

In [35], the simplex crossover (SPX) was proposed, a new multi-parent recombination operator for real-coded GAs. The experimental results with test functions used in their studies showed that SPX works well on functions having multimodality and/or epistasis with a medium number of parents: 3 parents on a low dimensional function or 4 parents on high dimensional functions. However, the authors did not consider the application of the SPX to multi-objective problems. Moreover, they did not consider the possibility to take into account the fitness of the objective function/s as the driving force of the simplex. Therefore, we decided to integrate in the GeDEA-II the SPX with these and further new distinctive features.

Before introducing the SPX exploited in the GeDEA-II, some words are spent to elucidate the Simplex algorithm, whose first release was presented in [17]. A simplex in n-dimensions is a construct consisting of n+1 solutions x_k, $k = 1, 2, \ldots, n + 1$ [18]. In a two dimensional plane, this corresponds to a triangle. The solutions are evaluated in each step and the worst solution $\mathbf{w}$, i.e., the one with the highest fitness value, is identified. The centroid, $\mathbf{M}$, of the remaining points is computed as $\mathbf{M} = \frac{1}{n}\sum_x x_k$ (k identifying the two best solutions) and a new solution $\mathbf{r}$, replacing $\mathbf{w}$, is obtained by *reflection*, $\mathbf{r} = \mathbf{M} + (\mathbf{M} - \mathbf{w})$. In the Nelder and Mead version of the simplex algorithm, further operators are considered, such as expansion, internal contraction and external contraction. However, they are not taken into account in this work, since this choice would result in additional functions evaluations, as well as add complexity to the algorithm. In Fig. 2, the reflection step of the Nelder and Mead simplex algorithm is depicted, applied to a problem in R^2.

$\mathbf{w}$ is the worst point, to be replaced by point $\mathbf{r}$. $\mathbf{M}$ is the centroid between the two other points, $\mathbf{x_1}$ and $\mathbf{x_2}$.

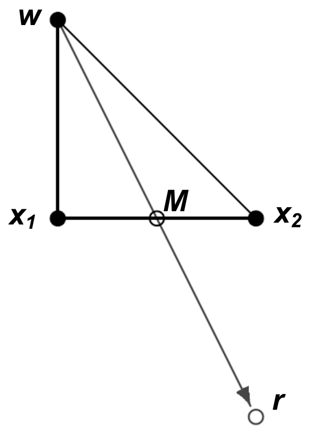


Fig. 2. The reflection step of the simplex algorithm applied to a problem in R^2.

GeDEA-II utilizes the Simplex concept as the crossover operator, in order to speed-up the evolution process, as stated below. As a matter of fact, crossover function plays an important role in the EAs, since it combines two individuals, or parents, to form a new individual, or child, for the next generation. Since the Simplex is itself an optimization algorithm, the generated children are expected to feature best fitness values when compared to the parents. Unlike the SPX presented in [35], the SPX exploited in GeDEA-II requires only two parents to form a new child. This choice was motivated by the following considerations. First of all, it is reminded here that two is the minimum number required to form a simplex. Second, from linear algebra, it can be easily demonstrated[1] that, given k vectors, there can be found more couples of mutually linearly independent vectors than can be done when considering triplets (or, in general, n-tuples) of independent vectors. As a straightforward consequence, it follows that this statement is even more true if not independence but only diversity (that is, at least one component different from a vector to another one) is required. Therefore, every time a new child is created, this characteristic of the SPX ensures that this child comprises genes different from those of the other children, and allows the greatest design space exploration, due to the diversity of the parents. These two parents are selected according to the selection procedure from the previous population, and combined following the guidelines of the simplex algorithm. Let assume ***p1***, ***p2*** being the two parent vectors, characterized by different, multiple fitness values, the child vector ***Child*** is formed according to the reflection move described in [18]:

$$\textbf{Child} := (1 + Refl) \cdot \mathbf{M} - Refl \cdot \mathbf{p_2} \qquad (1)$$

where ***Child*** is the new formed child and *Refl* is the reflection coefficient.

It is assumed that ***p1*** is the best fitness individual among the two chosen to form the ***Child***, whereas ***p2*** the worst one. Below the strategy followed to decide every time the best and the worst individual is highlighted.

[1]Let us consider three vectors in R^3 design space, namely $\vec{a}$=(1,0,0), $\vec{b}$=(0,1,0), and $\vec{c}$=(0,0,1). There can be found three couples of linearly independent vectors, that is [$\vec{a}$,$\vec{b}$], [$\vec{a}$,$\vec{c}$] and [$\vec{b}$,$\vec{c}$] but only a triplet of mutually and simultaneously independent vectors, that is the triplet [$\vec{a}$,$\vec{b}$,$\vec{c}$]. This simple demonstration remains valid when extended to the R^n space.

M is the centroid of ***p1***, calculated in the following manner:

$$\mathbf{M} := \left(\frac{1}{n}\right) \cdot (\mathbf{p_1}) \tag{2}$$

where n is the number of the remaining individual, excluded the worst one. A dedicated discussion will be done concerning this coefficient, and reported below. *Refl* coefficient is set equal to a random number ($refl \in [0, 1]$), unlike the elemental Simplex theory, which assumes a value equal to 1 for the *Refl* coefficient. This choice allows to create a child every time distant in a random manner from the parents, hence to explore more deeply the design space.

Moreover, unlike the Simplex algorithm theory, it was decided to switch from 1 to 2 the coefficient *n* in Eq. (2).

In order to establish if differences on performance exist between the two crossover configurations, that is the one with *n=1* and the one with *n=2*, minimization experiments on the test suite presented in V-A were performed. The GeDEA-II was executed equipped with the two crossover configurations 30 times, the population size and the number of generations being specified in Section V-B. The guidelines given in [36] were strictly followed. However, since these are multi-objective test problems, the best fitness value was replaced with the best hypervolume value (see Section V-C for more elucidations) of that particular run. Table I shows our results. The performance measures used are the following:

- A performance: average of the hypervolume indicator calculated at the end of each run;
- B performance: greater hypervolume indicator found;
- σ: standard deviation;

The Student parameter *T* in this table represents the result of a *t*-Test (at a 0.05 level of significance) as follows:

- *T* is introduced to ascertain if differences in A performance for the best crossover operator area significant when compared with the one of the other crossover configuration in the respective test problem.

In these columns, the crossover with the best A performance value is marked with ($**$), and the direction of any significant differences is denoted either by a ($+$) sign for an improvement in A performance or an approximate sign ($\cong$) if non-significant difference exists between the two averages.

TABLE I
STATISTICAL COMPARISON BETWEEN THE TWO SPX CONFIGURATIONS. RESULTS FOR THE *ZDT* AND *DTLZ* TEST SUITES.

	Coefficient *n* = 2			
	A	B	σ	T
ZDT1	1.1048	1.1063	0.0011	**
ZDT2	0.6204	0.6502	0.1100	**
ZDT3	1.1085	1.1103	0.0015	**
ZDT4	10.7612	11.1671	0.3928	**
ZDT6	7.4321	7.4441	0.0277	**
KUR	82.5799	83.2329	0.5201	*
DTLZ1	35.1480	35.7413	1.4350	**
DTLZ2	10.6261	10.6519	0.0171	**
DTLZ3	186.6339	188.2572	3.5181	**
DTLZ4	4.6273	4.6638	0.0255	**
DTLZ5	0.6398	0.6511	0.0048	**
DTLZ6	8.5188	8.5291	0.0112	**
DTLZ7	1.8662	1.9086	0.0221	**
	Coefficient *n* = 1			
	A	B	σ	T
ZDT1	0.7545	0.8627	0.0775	+
ZDT2	0.2492	0.3953	0.0746	+
ZDT3	0.9332	1.0037	0.0476	+
ZDT4	9.7198	10.4062	2.6585	$\cong$
ZDT6	4.2096	5.1145	0.5140	+
KUR	82.0038	83.5124	1.2317	$\cong$
DTLZ1	17.7558	26.4431	5.2643	+
DTLZ2	10.4655	10.5520	0.0568	+
DTLZ3	110.7428	156.1789	42.8868	+
DTLZ4	4.5029	4.6010	0.0619	+
DTLZ5	0.5673	0.6044	0.0214	+
DTLZ6	5.9904	8.5244	2.0642	+
DTLZ7	1.5801	1.9013	0.6277	$\cong$

First of all, the *n=2* configuration performs the best almost on all the test problems, and the standard deviation demonstrates that the repeatability of its performance is high level when compared to that of the other configuration.

Second, some test problems exist where the difference of the performance is not significant.

Third, the results analyzed as a whole show that when the coefficient *n* is 2, the SPX operator favors exploitation without penalizing exploration of the design space, and helps reaching the final approximation set, while covering it in a satisfactory manner.

Table II shows the percentages in which each crossover operator has obtained the best A performance on all the test functions. Its columns have the following meaning, according to the guidelines given in [36]:

- Best average/best *t*-test: percentage of test functions in which the crossover configuration has obtained the best A performance ($**$ in A column) and the application of the *t*-test confirms that it is significantly the best (plus sign ($+$) in the *T* column associated with the other crossover configuration); it is denoted with Roman number **I** in Table II.
- Best average/similar *t*-test: this column shows the same information as the previous one but the other crossover features no difference in A performance, according to the *t*-test (the other crossover configuration has a ($\cong$) sign in the *T* column); it is denoted with Roman number **II** in Table II.
- Total best: percentage of test functions in which the crossover achieves the best A performance, without considering the t-test. This percentage is calculated as the sum of the previous two columns; it is denoted with Roman number **III** in Table II.
- Similar *t*-test/no best average: percentage of test functions in which the crossover configuration shows, after the application of the *t*-test, non-significant differences in A performance regarding the best value (($\cong$) sign in the *T* column); it is denoted with Roman number **IV** in Table II.
- Total best/similar: percentage of test functions in which the crossover configuration achieves either the best A behavior or the one similar to the best. This percentage is the result of the sum of the two previous columns. it is denoted with Roman number **V** in Table II.

TABLE II
ANALYSIS FOR THE SPX OPERATORS PROVIDED WITH *n=2* AND *n=1* COEFFICIENT

	I	II	III	IV	V
Coefficient n=2	76.9 %	23.0 %	100 %	0 %	100 %
Coefficient n=1	0.0 %	0.0 %	0.0 %	23.1 %	23.1 %

Results presented in Table II confirm the preceding analysis, which shows the best performance achieved by *n=2* configuration.

The information about the fitness values is the key issue of this version of crossover: unlike the SPX operator presented in [35], this characteristic allows the crossover process to create a new individual, which is expected to be better than the parents. This new crossover operator was expected to combine both exploration and exploitation characteristics. In fact, the new formed child comprises the genes of two parents, that means a good exploration of the design space. However, it explores a design space region opposite to that covered by the parent number 2, that means it explores a region potentially not covered so far. In the early stages of the evolution, this means that child moves away from regions covered from bad parents, while exploring new promising ones.

Since the Simplex algorithm is itself a single-objective optimizer, a strategy was implemented to adapt it to a multi-objective algorithm. To deeply exploit the characteristics of the simplex, at each generation the mean of each objective function, extended to all the μ individuals, is computed. This mean is then compared to the one characterizing the previous generation, and the objective function featuring the greatest difference is selected as the fitness function used within the Simplex algorithm to decide every time the best and the worst individual. This choice was made after several experiments, which showed how a correct balance between exploration of the search space and the convergence to the P.F. can be achieved by means of a switching among multiple objective functions, each time selecting the most promising one.

Algorithm 1 presents the pseudo-code related to the application of the SPX in a multi-objective context, extended to the most general case involving M objective functions. It it assumed that all of the objectives are to be minimized. At each generation $ignr$, the mean of each objective function $mean$ is calculated. Based on these values, the percentage variations PV are subsequently derived. At this point, the two selected parents are sorted according to these values, and the child created according to Eqs. 1 and 2. This choice guarantees that the objective function characterized by the greatest difference is selected every time, therefore ensuring the highest convergence rate to the PF. For test problem involving more than two objective functions, the objective function considered to form the new child is chosen randomly in order to enhance the design space exploration of the crossover, required in highly dimensional objective spaces.

During evolution, GeDEA-II makes use exclusively of the SPX until half of the generations has been reached. After that, SPX is used alternatively with the SBX with a switching probability of 50 percent. This choice is motivated by the will of improving further the distribution and uniformity of the candidate solutions on the Approximate Pareto-optimal set.

[1]Here SM refers to the Shrink Mutation operator introduced in Section IV-B

[2]Hereafter *o.f.* stands for objective function

Algorithm 1 Application of SPX in a multi objective context.

```
1: $Set\ M = number\ of\ objectives$
2: $Set\ \mu = number\ of\ parents$
3: $Set\ ignr = current\ generation$
4: for $i = 1 \rightarrow M$ do
5:     $Mean = 0$
6:     for $j = 1 \rightarrow \mu$ do
7:         $MEAN(i)_{(ignr)} = Mean(i)_{(ignr)} + \sqrt{odfit(j)^2}$
8:     end for
9: end for
10: for $i = 1 \rightarrow M$ do
11:     $PV_i = \frac{MEAN(i)_{(ignr-1)} - MEAN(i)_{(ignr)}}{MEAN(i)_{(ignr-1)}}$
12: end for
13: $Set\ count = 1$
14: while $count <= \mu$ do
15:     $Choose\ two\ parents,\ \mathbf{p1}\ and\ \mathbf{p2},\ according\ to\ SM^1$
16:     for $i = 1 \rightarrow M$ do
17:         $oldfit(i,1) = i_{th}\ o.\ f.^2\ of\ parent\ \mathbf{p1}$
18:         $oldfit(i,2) = i_{th}\ o.\ f.\ of\ parent\ \mathbf{p2}$
19:     end for
20:     $Set\ A = max_{i \in M}\ (PV_i)$
21:     $Find\ index\ k \in M\ correspondent\ to\ A$
22:     $Set\ OLDFIT = [oldfit(k,1), oldfit(k,2)]$
23:     if $oldfit(k,1) \leq oldfit(k,2)$ then
24:         $Set\ M = \left(\frac{1}{n}\right) \cdot \mathbf{p1}$
25:         $\mathbf{Child} = (1 + Refl) \cdot M - Refl \cdot \mathbf{p2}$
26:     else
27:         $Set\ M = \left(\frac{1}{n}\right) \cdot \mathbf{p2}$
28:         $\mathbf{Child} = (1 + Refl) \cdot M - Refl \cdot \mathbf{p1}$
29:     end if
30:     $Set\ offpsring(count,:) = \mathbf{Child}$
31:     $count = count + 1$
32: end while
```

B. The Shrink Mutation

As far as mutation is concerned, a new Shrink-mutation operator is introduced in the GeDEA-II.

In the literature, this kind of mutation strategy is referred to as *Gaussian mutation* [37], and conventional implementations of Evolutionary Programming (EP) and Evolution Strategies (ES) for continuous parameter optimization using Gaussian mutations to generate offspring are presented in [26] and [38], respectively.

In general, mutation operator specifies how the genetic algorithm makes small random changes in the individuals in the population to create mutation children. Mutation provides genetic diversity and enables the genetic algorithm to search a broader space. Unlike the previous version of mutation featuring GeDEA algorithm, where some bits of the offspring were randomly mutated with a probability p_{mut}, here the mutation operator adds a random number taken from a Gaussian distribution with mean equal to the original value of each decision variable characterizing the entry parent vector.

The shrinking schedule employed is:

$$Shrink_i := Shrink_{i-1} \cdot \left(1 - \frac{ignr}{ngnr}\right) \tag{3}$$

where $Shrink_i$ is a vector representing the current mutation range allowed for that particular design variable, *ignr* represents the current generation and *ngnr* the total number of generations. The shape of the shrinking curve was decided after several experimental tests. The fact that the variation is zero at the last generation is also a key feature of this mutation operator. Being conceived in this manner, the mutation allows to deeply explore the design space during the first part of the optimization, while exploiting the non-dominated solutions during the last generations. Once the current variation range has been calculated, one decision variable of a selected child is randomly selected, and mutated according to the following formula:

$$Child_{mut} := Child_{cross} + [Shrink_i] \tag{4}$$

Unlike crossover operator, which generates all the offspring, mutation is applied only on a selected part of the offspring. Before starting offspring mutation, offspring population is randomly shuffled to prevent locality effects. After that, a pre-established percentage (fixed to 40% for all of the test problems) of the individuals are selected for mutation. The initial Shrink factor is set equal to the whole variation range of the design variables. This mutation operator was found to be powerful especially in multi-objective problems requiring a huge exploration of the design space.

C. Diversity preservation

As underlined in Sections 2 and 3, maintaining the genetic diversity within the population is mandatory for a robust EA. To this purpose, in GeDEA-II two diversity preservation mechanism are used, namely the GeDEM, already employed in GeDEA [11] and the Non-Dominated Sorting [8]. Both of the two mentioned mechanism are adopted since in authors' opinion each of them has unique features which can take benefit from each other. To make this assertion clearer, it is worth to briefly go back to the mathematical definition of GeDEM and non-dominated sorting based on crowding distance. The definition of dominance used in the non-dominated sorting procedure performed by GeDEM is:

Vector $\mathbf{u}$ = (rank$_u$; dist$_u$) dominates vector $\mathbf{v}$ = (rank$_v$; dist$_v$)
if and only if
$(\text{rank}_u > \text{rank}_v) \wedge (\text{dist}_u \geq \text{dist}_v)$

On the contrary, the definition of dominance used in the non-dominated sorting based on crowding distance is:

Vector $\mathbf{u}$ = (rank$_u$; dist$_u$) dominates vector $\mathbf{v}$ = (rank$_v$; dist$_v$)
if and only if
$(\text{rank}_u > \text{rank}_v) \vee [(\text{rank}_u = \text{rank}_v) \wedge (\text{dist}_u \geq \text{dist}_v)]$

Clearly, the logical operator is the great difference between the aforementioned diversity mechanisms, which entails the slightly different behavior of the two algorithms. In particular, GeDEM tends to create less non-dominated individuals, since both the rank and the diversity conditions are to be fulfilled simultaneously. Therefore, the evolution process results faster. On the other hand, non-dominated sorting based on crowding distance tends to create more non-dominated individuals, which results in a better Pareto front coverage. In order to take advantage of both the characteristics, in GeDEA-II the diversity preservation is accomplished by means of GeDEM, in the first three quarters of the generations, whereas in the remainder of the generations the Non-Dominated Sorting mechanism is exploited.

V. Comparison with Other Multiobjective Evolutionary Algorithms

In order to judge the performance of the GeDEA-II, a comparison with other different state-of-the-art multi-objective EAs was performed. SPEA-2 [10], NSGA-II [8] and IBEA [25] were chosen as competitors, and their performance against GeDEA-II was measured on two test problems featuring the characteristics that may cause difficulties in converging to the Pareto-optimal front and in maintaining diversity within the population [39]: discrete Pareto fronts, and biased search spaces. In addition, their performance was tested also on two more recent and more challenging benchmark test functions chosen among the scalable Test Problems presented in [40]. The four test functions, the methodology and the metric of performance used in the comparison are briefly recalled in the following for easy reference.

A. Test Functions

Here only four test problems are presented due to layout constraints. The original version of *ZDT*$_3$ and *ZDT*$_6$ presented in [4] featured 30 and 10 decision variables, respectively. Here we propose them with 100 decision variables. As regards *DTLZ*$_3$, the number of variables suggested in [40] is 12. Here we propose it with 22 decision variables, respectively. As regards *DTLZ*$_7$, we increased the number of decision variables from the original one equal to 22, up to 100.

B. Methodology

The methodology used in [4] is strictly followed. GeDEA-II and competitors are executed 30 times on each test function. There are different parameters associated with the various algorithms, some common to all and some specific to a particular one. In order to make a fair comparison among all the algorithms, most of these constants are kept the same. In GeDEA-II, GeDEA and in competitors' algorithms, the population size is set to 100. In the following, the parameters of the competitors MOEA are reported following the terminology used in PISA implementation[2]. The individual mutation

[2]*Individual mutation probability* (probability that a certain individual undergoes mutation); *individual recombination probability* (probability that a certain pair of individuals undergoes recombination); *variable mutation probability* (probability that a certain variable in a given individual is mutated); *variable swap probability* (probability that a certain pair of variables is swapped during recombination); *variable recombination probability* (probability that the SBX recombination operator is used for a given pair of variables; this decision is independent from variable swap probability); $\eta_{mutation}$ (distribution index for mutation operator); $\eta_{recombination}$ (distribution index for recombination operator).

probability is always 1 and the variable mutation probability is fixed at $1/n$, n being the number of the decision variables of the test problem considered. The individual recombination probability along with the variable recombination probability are set to 1. The variable swap probability is set to 0.5. $\eta_{mutation}$ is always set to 20 and $\eta_{recombination}$ is fixed to 15. For IBEA algorithm, tournament size is always set to 2, whereas additive epsilon is chosen as the indicator. Scaling factor $kappa$ is set to 0.05, and rho factor is fixed to 1.1. For both NSGA-II and SPEA2, tournament size is given a value equal to 2. NSGA-II, SPEA2 and IBEA are run with the PISA[3] implementation [41], with exactly the same parameters and variation operators. The number of generations was intentionally reduced in order to test the convergence properties of the investigated algorithms, and contribute to justify the different results reported here, when compared to those presented in the original papers [4], [40].

TABLE III
ORIGINAL AND PROPOSED NUMBER OF GENERATIONS FOR THE *ZDT* AND *DTLZ* TEST PROBLEMS.

	Number of generations	
	Original version problems	Proposed test problems
ZDT3	250	40
ZDT6	250	30
DTLZ3	500	150
DTLZ7	200	100

C. Metric of Performance

Different metrics can be defined to compare the performance of EAs with respect to the different goals of optimization itself [4]: how far is the resulting non-dominated set from the Pareto front, how uniform is the distribution of the solutions along the Pareto approximation set/front, how wide is the Pareto approximation set/front. For measuring the quality of the results, we have employed the hypervolume approach, due to its construction simplicity and for the reason, which will be soon explained. The hypervolume approach presented in [42] (modified in [10]) measures how much of the objective space is dominated by a given nondominated set. Zitzler et al. state it as the most appropriate scalar indicator since it combines both the distance of solutions (towards some utopian trade-off surface) and the spread of solutions. To better understand the reason for this choice, it is worth to see at Fig. 3 (a) and (b). The reference point is indicated as R.P..

As regards the convergence of the known P.F. to the True P.F., please consider the case depicted in Fig. 3 (a). The non-dominated set A has a great hypervolume indicator when compared to set B, due to its superior proximity to the True P.F.. As far as the spread of the solution on the Pareto approximation set is concerned, Fig. 3 (b) qualitatively shows that a more uniform distribution of the solutions (on the right) yields a greater hypervolume indicator. Therefore, this indicator is intrinsically able to compare performance

[3]This software is available for public use at PISA website http://www.tik.ee.ethz.ch/pisa/

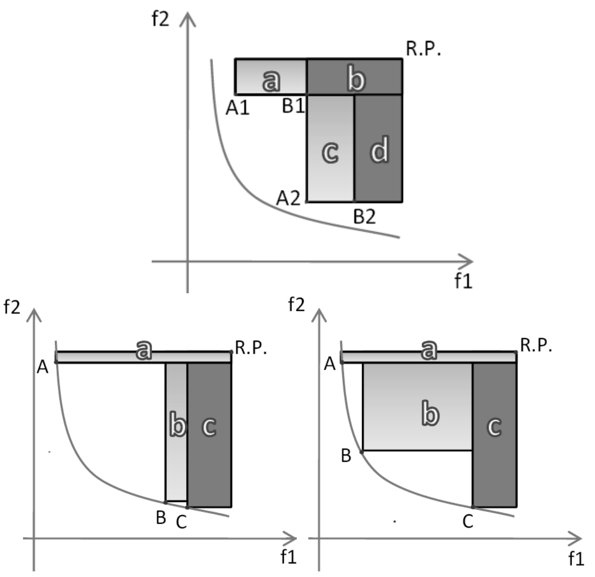


Fig. 3. Significance of the hypervolume indicator as far as convergence (a, AT THE TOP), and diversity (b, AT THE BOTTOM) is concerned.

of different EAs as regards both the convergence to the Pareto approximation set and its coverage. In general, it is not sufficient for a set of candidate solutions to be closer than another one to the True Pareto front, to have a higher hypervolume value. It is the blend of convergence and uniformity of the final approximation set that counts.

The hypervolume[4] is defined as the area of coverage of PF_{known} with respect to the objective space for a two-objective MOP. As illustrated in Fig. 3, this region consists of an orthogonal polytope, and may be seen as the union of n axis-aligned hyper-rectangles with one common vertex (the reference point, R.P.). Mathematically, this is described in Eq. (5) (for a generic n-objectives problem):

$$hypervolume := \left[\bigcup_i vol_i | vec_i \in P_{known} \right] \tag{5}$$

where vec_i is a nondominated vector in PF_{known} and vol_i is the hypervolume (an area in two objectives problems) between the reference point and vector vec_i.

In this work, the version implemented by Fonseca et al. and presented in [43] is adopted.

D. Results of Comparison

As in Zitzler et al. [4], Figures 4, 6 and 7 show an excerpt of the non-dominated fronts obtained by the EAs and the Pareto-optimal fronts (continuous curves). The points plotted are the non-dominated solutions extracted from the union set of the outcomes of the first five runs, the best and the worst one being discarded. The performance of GeDEA-II is also compared to that of the competitors according to the hypervolume metric as defined in [43]. The distribution of these values is shown using box plots in Figures 5 and 8. On each box, the central line represents the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually, with a Plus sign. Results

[4]The hypervolume is a Pareto compliant indicator as stated in [23].

are normalized with the best Hypervolume value coming from the union set of all of the runs, extended to all of the algorithms. For each test problem, the reference point is assumed equal for all of the algorithms, and equal to the maximum value for each objective function from the union of all of the output points.

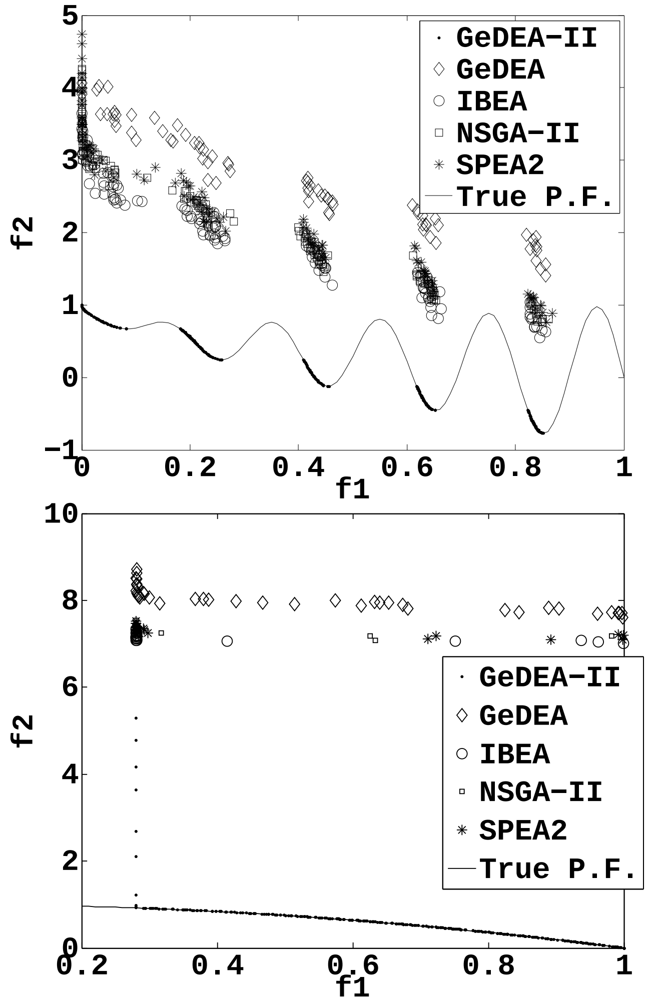


Fig. 4. Test functions ZDT_3 (AT THE TOP) and ZDT_6 (AT THE BOTTOM).

In general, the experimental results show that GeDEA-II is able to converge towards the True Pareto-optimal front and to develop a widely and well distributed non-dominated set of solutions. The comparison with the other three best-performing MOEAs according to the *Hypervolume* metric proves that the performance of GeDEA-II is somewhat superior. Considering the specific features of the two *ZDT* test functions, GeDEA-II shows similar performance both on multi-front and biased Pareto-optimal fronts. NSGA-II, SPEA-2 and IBEA seem instead to have more difficulties with discreteness (test function ZDT_3). The performance of GeDEA-II is particularly remarkable in the case of biased search space (test function ZDT_6) where it is also able to evolve a well-distributed non-dominated set. These results gain even more significance, since the number of decision variables was set to 100, unlike the original values of 30 (10 for the test function ZDT_6).

As far as $DTLZ_3$ and $DTLZ_7$ test functions is concerned, GeDEA-II is able to reach the True Pareto Front, whereas the competitors remain trapped in the local Pareto Approximation Sets, as shown in Fig. 6 and 7.

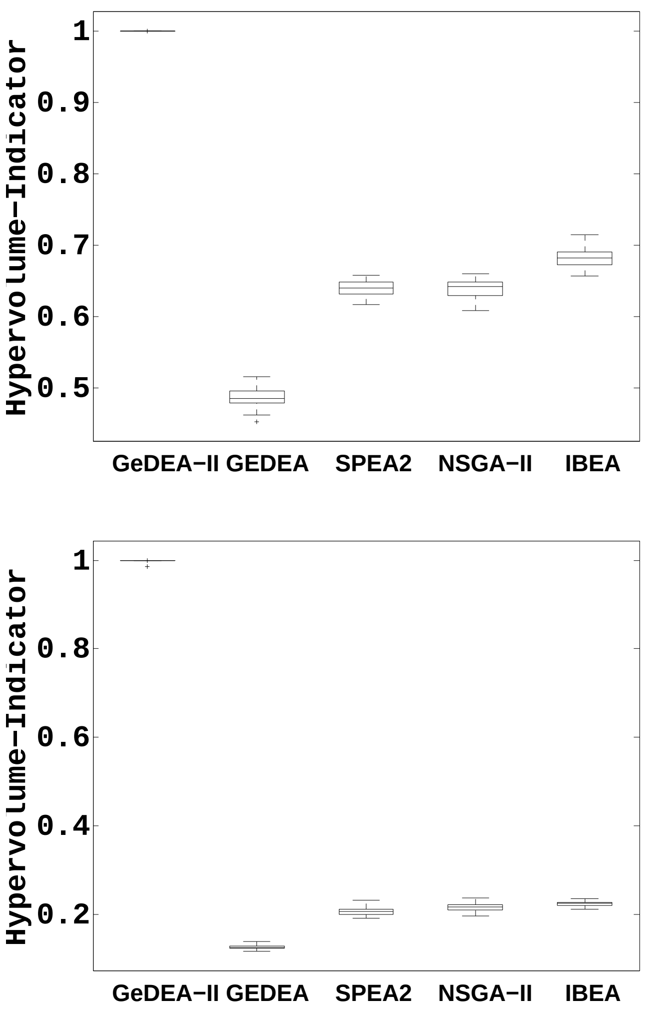


Fig. 5. Box plots based on the *Hypervolume* metric. Each square contains six box plots representing the distribution of *Hypervolume* values for the six algorithms. Results refer to the ZDT_3 (AT THE TOP) and ZDT_6 (AT THE BOTTOM) test functions.

Finally, box plots prove, in general, that the performance of GeDEA-II is superior to those of the competitors also as far as the repeatability of the results is concerned.

E. GeDEA-II Performance on Extremly Multidimensional Landscapes

In this section, authors aim at putting in evidence the outstanding performance of GeDEA-II even on high multidimensional environments. To do this, two test problems, chosen among those presented in Section V-A are considered, and the GeDEA-II performance tested by changing every time the number of decision variables. Test functions chosen for this test are the ZDT_4 and $DTLZ_3$, that is, the most difficult to solve problems, as stated in [4] and [40].

In Table IV, the number of variables and generations characterizing these tests are reported. In particular, ZDT_4 test function feature a maximum number of decision variables of 1000, whereas on $DTLZ_3$ test functions the maximum number of decision variables is increased up to 100 times the original proposed number [40].

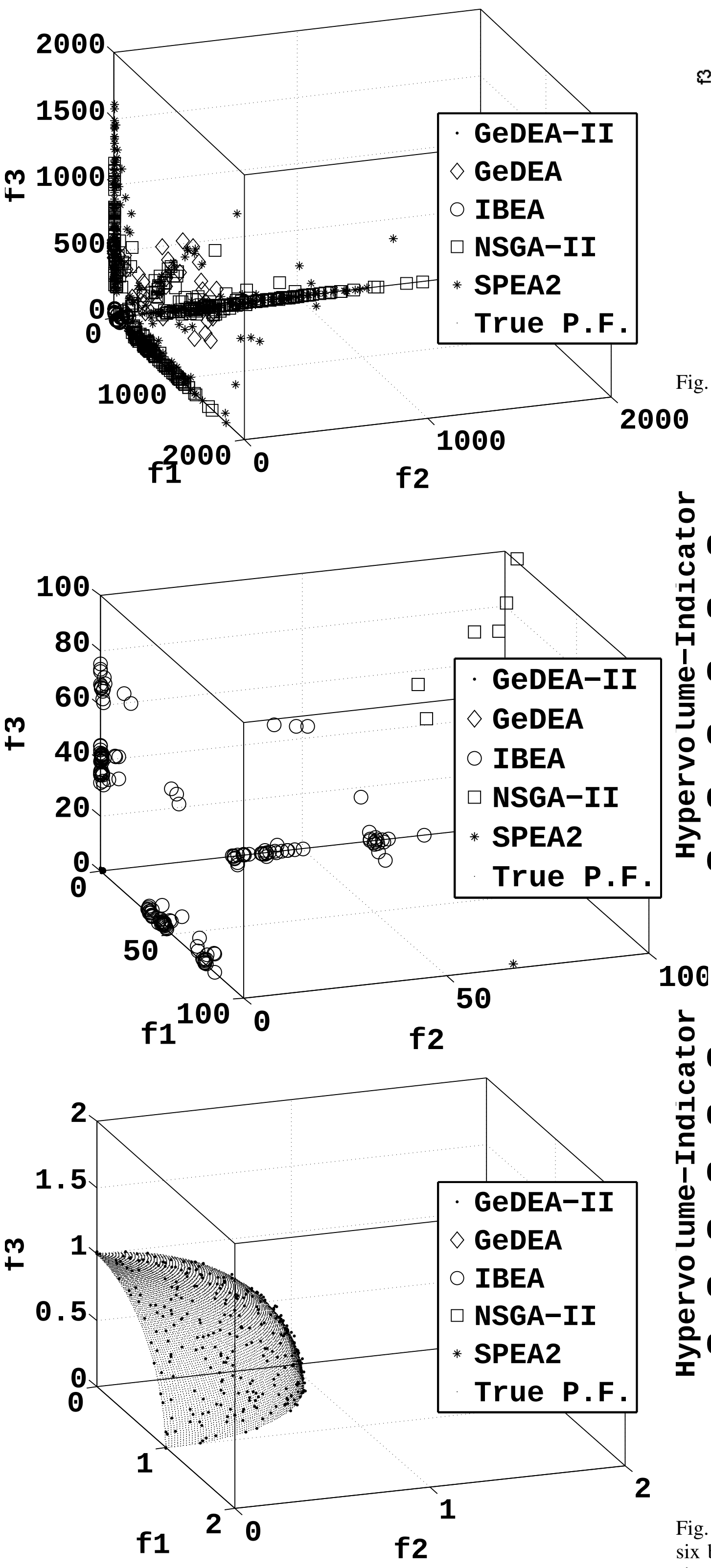


Fig. 6. Test function *DTLZ3*. From the left, Auto scale axes, Medium zoom and True Pareto Front region.

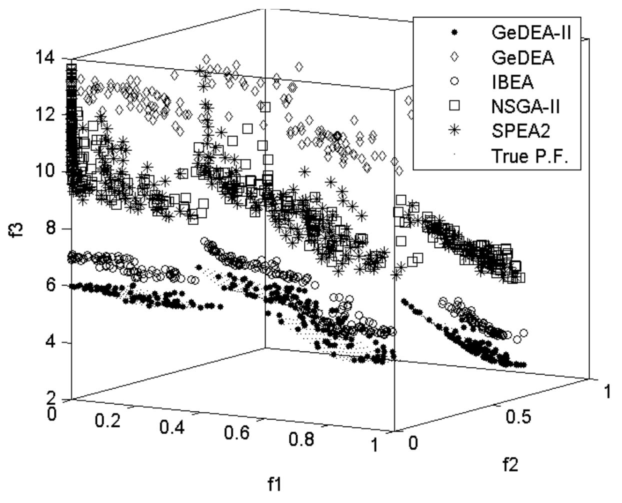


Fig. 7. Test function $DTLZ_7$.

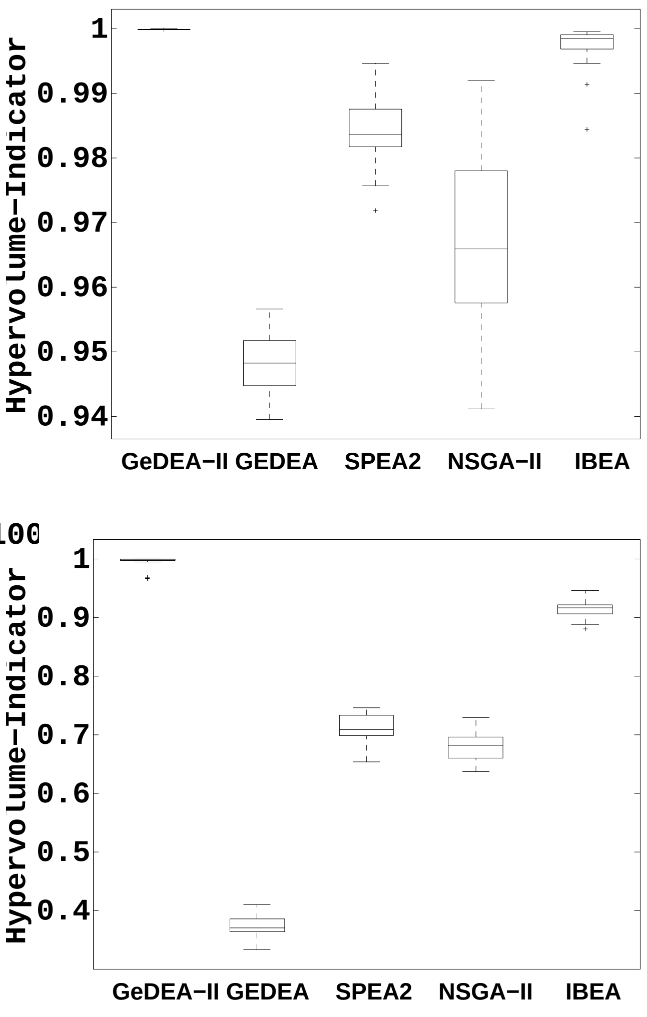


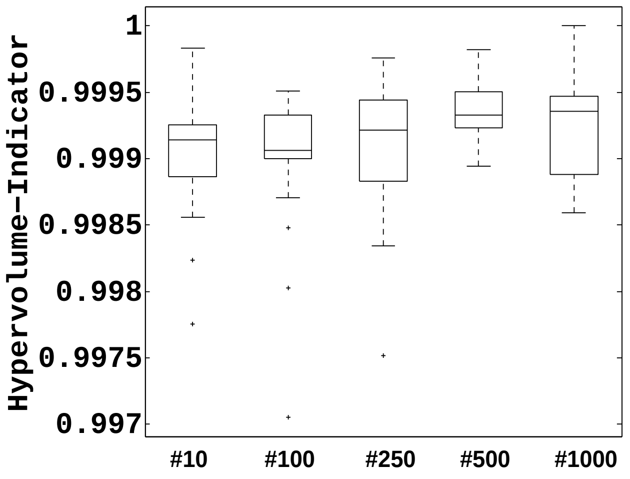
Fig. 8. Box plots based on the $Hypervolume$ metric. Each square contains six box plots representing the distribution of $Hypervolume$ values for the six algorithms. Results refer to the $DTLZ_1$ (AT THE TOP) and $DTLZ_7$ (AT THE BOTTOM) test functions.

To the best of the authors' knowledge, this is the first time a MOEA is tested on these test problems, with these number of decision variables. For each test problems, we

TABLE IV
MINIMUM AND MAXIMUM NUMBER OF DECISION VARIABLES FOR THE ZDT_4 and $DTLZ_3$ test problems.

	Number of generations	Minimum number of decision variables	Maximum number of decision variables
ZDT4	40	10	1000
DTLZ3	80	12	1200

performed 30 independent runs for each number of decision variables, and the boxplots were then built, following the guidelines already given in Section V-D. Y-axes are scaled in such a way the best run is given a value equal to 1. In Figure 9, the boxplots showing GeDEA-II performance are presented, as the decision variables are increased from the minimum value up to the maximum one. Results clearly



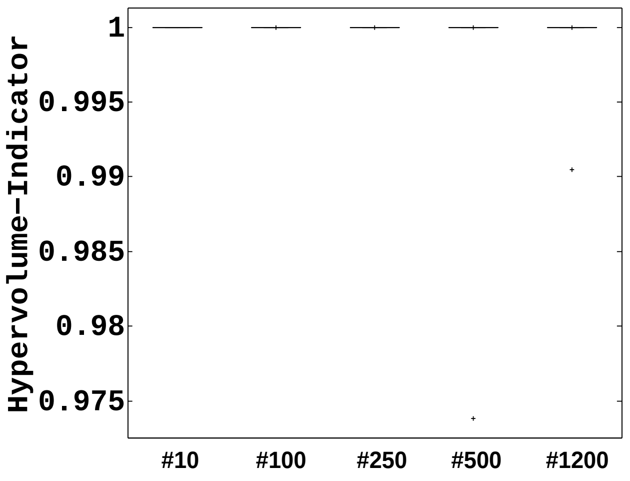
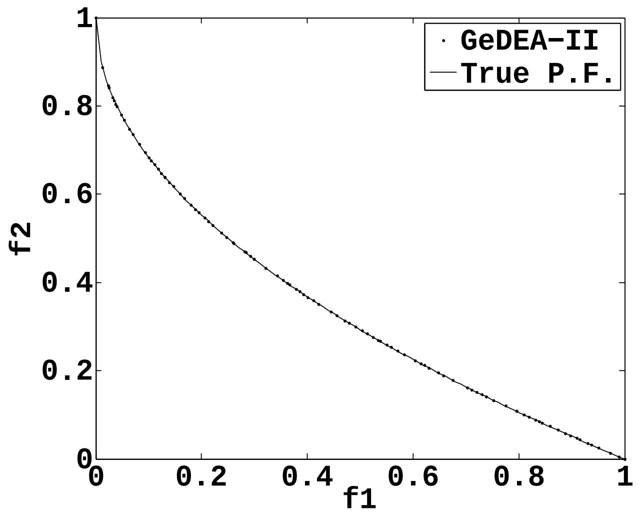


Fig. 9. Box plots based on the *Hypervolume* metric. Each square contains five box plots representing the distribution of *Hypervolume* values for the six number of decision variables.Results refer to the ZDT_4 (AT THE TOP) and $DTLZ_3$ (AT THE BOTTOM) test functions.

states that GeDEA-II performance is high-level. In each test problem, performance is never lower than 99% of the maximum value, no matter how many the decision variables are. This clearly demonstrate GeDEA-II manages to evolve the initial population near to the True Pareto front, even when



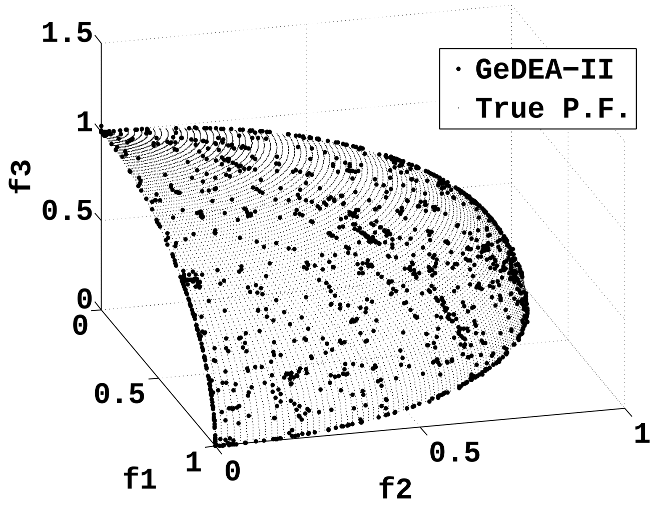


Fig. 10. Final Approximation Set reached by the GeDEA-II on test function ZDT_4 (AT THE TOP) and the non dominated solutions found by GeDEA-II on $DTLZ_3$ (AT THE BOTTOM), featuring 1200 decision variables.

the number of decision variables is dramatically increased. Figure 10 shows in the objective space, the distribution of the final solutions obtained in the run with the lowest Hypervolume-value by the GeDEA-II for each test instance, for the maximum number of decision variables. It is evident that as regards the convergence to the True Pareto Front and spread of solutions, GeDEA-II performance is high level.

VI. CONCLUSION

In this paper, we have presented GeDEA-II, an improved multi-objective evolutionary algorithm that employs novel variation operators compared to its predecessor GeDEA. Extensive numerical comparisons of GeDEA-II with GeDEA and with NSGAII, SPEA-2 and IBEA, three state-of-the-art recently proposed algorithms, have been carried out on various test problems. Moreover, optimization difficulties have been enhanced further, in order to test the robustness of the codes. The key results of the comparison show the excellent performance of the GeDEA-II, when compared to the competitors algorithm, in terms of both exploration and exploitation capabilities. Boxplots show that the reproducibility of results of GeDEA-II is high-level, when compared to that of the NSGAII, SPEA-2 and IBEA. In extremely high dimensional spaces, GeDEA-II clearly shows excellent performance. In addition to these characteristics, GeDEA-II performs these tasks with a reduced number of objective functions evaluations, a very useful feature when considering its application to real-world engineering problems.

APPENDIX

Each of the three *ZDT* test functions, namely *ZDT*$_3$, *ZDT*$_4$ and *ZDT*$_6$ introduced in [4] is a two-objective minimization problem that involves a distinct feature among those identified in [39]. All the test functions are constructed in the same way, according to the guidelines in [39]:

$$\begin{aligned} Minimize: T(\mathrm{x}) &= (f_1(x_1), f_2(x)) \\ subject\ to: f_2(\mathrm{x}) &= g(x_2;\ldots;x_m) \\ & h(f_1(x_1), g(x_2;\ldots;x_m)) \\ where: \mathrm{x} &= (x_1,\ldots,\mathrm{x_M}) \end{aligned} \tag{6}$$

Function *f* controls vector representation uniformity along the Pareto approximation set. Function *g* controls the resulting MOP characteristics (whether it is multifrontal or has an isolated optimum). Function *h* controls the resulting Pareto front characteristics (e.g., convex, disconnected, etc.) These functions respectively influence search along and towards the true Pareto front, and the shape of a Pareto front in R^2. Deb [39] implies that a MOEA has difficulty finding PF$_{true}$ because it gets "trapped" in the local optimum, namely PF$_{local}$. Test functions reported in this work feature an increased number of decision variables, when compared to their original versions reported in [4]. This choice was motivated by the authors' will of testing exploration capabilities of the algorithms also on highly dimensional test problems, and contributes to justify the results presented in Section V-D.

- Test function *ZDT*$_3$ features a Pareto-optimal front disconnected, consisting of several noncontiguous convex parts:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1 + 9\cdot\sum_{i=2}^{n}\frac{x_i}{(n-1)} \\ h(f_1,g) &= 1 - \left(\sqrt{\frac{f_1}{g}}\right) - \left(\frac{f_1}{g}\right)\cdot\sin(10\pi f_1) \end{aligned} \tag{7}$$

  where *n* = 100 and $x_i \in$ [0,1]. The Pareto-optimal front corresponds to *g*(x) = 1. The original version presented in [4] featured 30 decision variables.
- Test function *ZDT*$_4$ contains 21^9 local Pareto-optimal fronts and, therefore, tests for the EA ability to deal with multifrontality:

$$\begin{aligned} f_1(x_1) &= (x_1) \\ g(x_2;\ldots;x_n) &= 1 + 10(n-1) \\ & \cdot\sum_{i=2}^{n}\left({x_i}^2 - 10\cos(4\pi x_i)\right) \\ h(f_1,g) &= 1 - \sqrt{\frac{f_1}{g}} \end{aligned} \tag{8}$$

  where *n* = 100 and $x_i \in$ [0,1]. The Pareto-optimal front is convex and corresponds to *g*(x) = 1. The original version presented in [4] featured 10 decision variables.
- Test function *ZDT*$_6$ features two difficulties caused by the non-uniformity of the search space: first, the Pareto optimal solutions are nonuniformly distributed along the PF$_{true}$ (the front is biased for solutions for which $f_1(x_1)$ is near one); and second, the density of the solutions is lowest near the PF$_{true}$ and highest away from the front::

$$\begin{aligned} f_1(x_1) &= 1 - \exp(-4x_1)\sin^6(6\pi x_1) \\ g(x_2;\ldots;x_n) &= 1 + 9\cdot\left(\sum_{i=2}^{n}\frac{x_i}{(n-1)}\right)^{1/4} \\ h(f_1,g) &= 1 - \left(\frac{f_1}{g}\right)^2 \end{aligned} \tag{9}$$

  where *n* = 100 and $x_i \in$ [0,1]. The Pareto-optimal front is non-convex and corresponds to *g*(x) = 1. The original version presented in [4] featured 10 decision variables.

Finally, two of the tri-objective minimization test functions designed by Kalyanmoy Deb, Lothar Thiele, Marco Laumanns and Eckart Zitzler, and presented in [40], are considered, in order to demonstrate the GeDEA-II capabilities on more than two-objectives test problems. In the following, *n* identifies the number of decision variables, *M* the number of objective functions, and $k = |x_M| = n - M + 1$ the number of variables of the functional $g(x_M)$. The number of variables was always increased when compared to that suggested by the authors in [40], whereas the decision variables range was left unchanged. These features help clarifying the different results between those reported in Section V-D and the original ones [40].

- Test function *DTLZ*$_3$ is similar to test function *DTLZ*$_2$, except for the function g, which introduces $(3^k - 1)$ local Pareto-optimal fronts, and only one global Pareto-optimal front.

$$\begin{aligned} f_1(x) &= (1 + g(x_M))\cos(x_1\pi/2)\cos(x_2\pi/2) \\ f_2(x) &= (1 + g(x_M))\cos(x_1\pi/2)\sin(x_2\pi/2) \\ f_3(x) &= (1 + g(x_M))\sin(x_1\pi/2) \\ g &= 100\cdot[k + \sum_{x_i\in x_M}(x_i - 0.5)^2 - \\ & \cos(20\pi\,(x_i - 0.5))] \end{aligned} \tag{10}$$

  where *n* = 22 and $x_i \in$ [0,1]. The number of variables suggested in [40] is 12.
- Test function *DTLZ*$_7$ features 2^{M-1} disconnected local Pareto-optimal regions in the search space. It is chosen to test the MOEA ability in finding and maintain stable and distributed subpopulations in all four disconnected global Pareto-optimal regions.

$$\begin{aligned} f_1(x) &= x_1 \\ f_2(x) &= x_2 \\ f_3(x) &= (1 + g(x_M))h \\ g &= 1 + \frac{9}{k}\sum_{x_i\in x_M}(x_i) \\ h &= M - \sum_{i=1}^{M-1}\left[\frac{f_i}{1+g}\left(\sin((1 + 3\pi f_i))\right]\right. \end{aligned} \tag{11}$$

where n = 100 and $x_i \in$ [0,1]. Once again, the number of decision variables was dramatically increased when compared to the original one, suggested in [40] for this test problem, and equal to 22.

References

[1] C. M. Fonseca and P. J. Fleming, *Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization.* In S. Forrest (Ed.), Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, California, pp. 416-423. Morgan Kaufmann., 1993.

[2] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *In Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, 1994, pp. 82–87.

[3] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[4] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[5] E. Zitzler and L. Thiele, *An evolutionary algorithm for multiobjective optimization: The strength pareto approach.* Technical Report 43, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland., 1998.

[6] ——, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271., 1999.

[7] J. Knowles and D. Corne, "The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation," *IEEE Press*, vol. 1, pp. 98–105, 1999.

[8] 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, 2002.

[9] D. W. Corne, J. D. Knowles, and M. J. Oates, *The pareto envelope-based selection algorithm for multiobjective optimisation.* Proceedings of the Parallel Problem Solving from Nature, VI Conference, Berlin, pp. 839-848. Springer., 2000.

[10] E. Zitzler, M. Laumanns, and L. Thiele, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm.* Technical Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland,May 2001., 2001.

[11] A. Toffolo and E. Benini, "Genetic diversity as an objective in multi-objective evolutionary algorithms," *Evolutionary Computation*, vol. 11, no. 2, pp. 151–157, 2002.

[12] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: model, taxonomy and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474–488, 2005.

[13] N. Krasnogor, W. Hart, and J. Smith, *Recent Advances in Memetic Algorithms, Studies in Fuzziness and Soft Computing.* Springer-Verlag, 2004.

[14] P. Moscato, R. Berretta, and C. Cotta, *Memetic Algorithms.* John Wiley Sons, Inc., 2010.

[15] K. Deb, S. Lele, and R. Datta, "A hybrid evolutionary multi-objective and sqp based procedure for constrained optimization," *Advances in Computation and Intelligence*, vol. 4683, pp. 36–45, 2007.

[16] H. Ghiasi, D. Pasini, and L. Lessard, "A non-dominated sorting hybrid algorithm for multi-objective optimization of engineering problems," *Engineering Optimization*, vol. 43, no. 1, pp. 39–59, 2011.

[17] W. Spendley, G. R. Hext, and F. R. Himsworth, "Sequential Application of Simplex Designs in Optimization and Evolutionary Operation," *Technometrics*, vol. 4, pp. 441–461, 1962.

[18] J. M. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, 1965.

[19] R. Chelouah and P. Siarry, "Genetic and nelder-mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions," *European Journal of Operational Research*, vol. 148, no. 2, pp. 335–348, 2003.

[20] S.-K. S. Fan, Y.-C. Liang, and E. Zahara, "A genetic algorithm and a particle swarm optimizer hybridized with nelder-mead simplex search," *Computers Industrial Engineering*, vol. 50, pp. 401–425, 2006.

[21] X. Guo, "A hybrid simplex multi-objective evolutionary algorithm based on preference order ranking," in *Computational Intelligence and Security, International Conference on*, 2011, pp. 29–33.

[22] K. Deb, *Multi-objective Optimization using Evolutionary Algorithms.* John Wiley, Chichester, UK., 2001.

[23] C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-objective Problems.* Kluwer Academic Publishers, New York, NY, 2002.

[24] P. Koduru, Z. Dong, S. Das, S. Welch, J. L. Roe, and E. Charbit, "A multiobjective evolutionary-simplex hybrid approach for the optimization of differential equation models of gene networks," *IEEE Trans. Evolutionary Computation*, vol. 12, no. 5, pp. 572–590, 2008.

[25] E. Zitzler and S. Künzli, *Indicator-Based Selection in Multiobjective Search.* In *Parallel Problem Solving from Nature (PPSN VIII)*, X. Yao et al., Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 832-842., 2004.

[26] T. Bäck, *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, 1996.

[27] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, Reading, Massachusetts., 1989.

[28] R. B. Agrawal and K. Deb, "Simulated binary crossover for continuous search space," *Complex Systems, 9, pp. 115-148.*, 1994.

[29] K. Deb and M. A. Goyal, "Combined Genetic Adaptive Search (GeneAS) for Engineering Design," *Computer Science and Informatics*, vol. 26, pp. 30–45, 1996.

[30] T. Bäck, "Selective pressure in evolutionary algorithms: A characterization of selection mechanisms," *In Proceedings of the First IEEE Conference on Evolutionary Computation*, pp. 57–62, 1994.

[31] C. Comis Da Ronco and E. Benini, "Gedea-II: A novel evolutionary algorithm for multi-objective optimization problems based on the simplex crossover and the shrink mutation," in *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering and Computer Science 2012, WCECS 2012, 24-26 October, 2012, San Francisco, USA*, pp. 1298–1303.

[32] A. E. Eiben and T. Bäck, "Empirical investigation of multiparent recombination operators in evolution strategies," *Evolutionary Computation*, vol. 5, no. 3, pp. 347–365, 1997.

[33] I. Ono and S. Kobayashi, "A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover," *Proceedings of the Seventh International Conference on Genetic Algorithms*, pp. 246–253, 1997.

[34] S. Tsutsui and A. Ghosh, "A study on the effect of multi-parent recombination in real coded genetic algorithms," *Proceedings of the 1998 IEEE ICEC*, pp. 828–833, 1998.

[35] S. Tsutsui, M. Yamamura, and T. Higuchi, "Multi-parent recombination with simplex crossover in real coded genetic algorithms," *Proceedings of the GECCO-99*, pp. 657–644, 1999.

[36] F. Herrera, M. Lozano, and A. M. Sanchez, "A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study," *International Journal of Intelligent Systems*, vol. 18, pp. 309–338, 2003.

[37] T. Bäck and H.-P. Schwefel, "An overview of evolutionary algorithms for parameter optimization," *Evolutionary Computation*, vol. 1, no. 1, pp. 1–23, 1993.

[38] D. B. Fogel, *Evolutionary computation: toward a new philosophy of machine intelligence.* Piscataway, NJ, USA: IEEE Press, 1995.

[39] K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, 1999.

[40] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multi-objective optimization," *Computer Engineering and Networks Laboratory (TIK), TIK-Technical Report No. 112, Swiss Federal Institute of Technology*, 2001.

[41] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler, *PISA - A Platform and Programming Language Independent Interface for Search Algorithms.* Springer, 2002.

[42] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms on test functions of different difficulty," *Proceedings of the 1999 Genetic and Evolutionary Computation Conference*, pp. 121–122, 1999.

[43] C. M. Fonseca, L. Paquete, and M. López-Ibáñez, "An improved dimension-sweep algorithm for the hypervolume indicator," *IEEE Congress on Evolutionary Computation*, pp. 1157–1163, 2006.