Coevolutionary Operations for Large Scale Multi-objective Optimization

1st Luis Miguel Antonio
Artificial Intelligence Department
GO-SHARP
Mexico City, Mexico
luis.miguel@go-sharp.com

2nd Carlos A. Coello Coello
Computer Science Department
CINVESTAV-IPN
Mexico City, Mexico
ccoello@cs.cinvestav.mx

3rd Mario A. Ramírez Morales
Technological Innovation Department
CIDETEC-IPN
Mexico City, Mexico
mramirezmo@ipn.mx

4th Silvia González Brambila
Computer Science Department
UAM Azcapotzalco
Mexico City, Mexico
sgb@azc.uam.mx

5th Josué Figueroa González
Computer Science Department
UAM Azcapotzalco
Mexico City, Mexico
jfgo@azc.uam.mx

6th Guadalupe Castillo Tapia
Administration Department
UAM Azcapotzalco
Mexico City, Mexico
mgct@correo.uam.mx

***Abstract*—Multi-objective evolutionary algorithms (MOEAs) of the state of the art are created with the only purpose of dealing with the number of objective functions in a multi-objective optimization problem (MOP) and treat the decision variables of a MOP as a whole. However, when dealing with MOPs with a large number of decision variables (more than 100) their efficacy decreases as the number of decision variables of the MOP increases. On the other hand, problem decomposition, in terms of decision variables, has been found to be extremely efficient and effective for solving large scale optimization problems. Nevertheless, most of the currently available approaches for large scale optimization rely on models based on cooperative coevolution or linkage learning methods that use multiple subpopulations or preliminary analysis, respectively, which is computationally expensive (in terms of function evaluations) when used within MOEAs. In this work, we study the effect of what we call operational decomposition, which is a novel framework based on coevolutionary concepts to apply MOEAs's crossover operator without adding any extra cost. We investigate the improvements that NSGA-III can achieve when combined with our proposed coevolutionary operators. This new scheme is capable of improving efficiency of a MOEA when dealing with large scale MOPs having from 200 up to 1200 decision variables.**

***Index Terms*—Bio-inspired optimization, large scale multi-objective optimization, decomposition, multi-objective optimization**

I. Introduction

There exist many problems in several disciplines that require the optimization of multiple objective functions at the same time. They are called *multi-objective optimization problems* (MOPs), and their solution involves finding the best possible trade-offs among their objectives. This set of trade-off solutions is called the *Pareto optimal set*, and their corresponding objective function values form the so-called *Pareto front*. Obtaining the Pareto front is the main goal in multi-objective optimization. MOPs have been solved during many years, using mathematical programming techniques [1]. However, the fact that a wide variety of MOPs in real-world applications tend to be nonlinear, and perhaps even non-differentiable, has motivated the use of metaheuristics to deal with them. From the many metaheuristics in current use, Evolutionary Algorithms (EAs) are the most popular choice in the specialized literature since they are considered to be among the best multi-objective optimizers [2]. Multi-objective evolutionary algorithms (MOEAs) have the advantage of being population-based, which allows them to generate several elements of the Pareto optimal set in a single run, whereas mathematical programming techniques usually produce a single element per run. Additionally, MOEAs are less sensitive to the continuity and shape of the Pareto front, while these features normally represent serious difficulties for mathematical programming techniques.

The current practice of MOEAs is to assess their performance using benchmark problems such as the Zitzler-Deb-Thiele (ZDT) [3], the Deb-Thiele-Laumanns-Zitzler (DTLZ) [4] and the Walking-Fish-Group (WFG) [5] test suites, which are normally adopted with a relatively low number of decision variables (usually, adopting a maximum of up to 30 decision variables). However, in real-world applications, many MOPs have hundreds or even thousands of decision variables and the effect of parameter scalability in modern MOEAs has not been properly analyzed. In fact, scalability in decision variable space is a topic that has been only scarcely studied in the context of MOEAs. This is perhaps motivated by the fact that most researchers assume that the currently available MOEAs should be able to work properly with a large number of decision variables. Nevertheless, there exists empirical evidence that indicates that most of the currently available MOEAs significantly decrease their efficacy as the number of decision variables of a MOP increases [6], [7]. The work reported here tries to contribute to this important topic.

In this paper, we propose a new scheme to apply a MOEAs'

The second author gratefully acknowledges support from CONACyT project no. 1920 and from a 2018 SEP-Cinvestav grant (application no. 4).

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

crossover operators which improves the performance over large scale MOPs. We study here the effect of parameter scalability and investigate the improvements that a MOEA can achieve when adopting this scheme. For this purpose, we propose to combine the NSGA-III [8] with Cooperative Coevolutionary techniques (which have shown to be very effective for large scale single-objective optimization [9], [10]), giving rise to a novel MOEA based on a new framework inspired by coevolutionary operations.

The remainder of this paper is organized as follows. Section II states the problem of our interest. The previous related work is discussed in Section III. Section IV describes our proposed approach and the experiments carried out to validate it. Finally, our conclusions and some possible paths for future work are drawn in Section V.

II. The Multi-objective Optimization Problem

Formally, a multi-objective optimization problem (MOP) is defined as:

$$\text{minimize} \quad f(x) = [f_1(x), f_2(x), \ldots, f_k(x)]^T \tag{1}$$

subject to:

$$g_i(x) \leq 0 \qquad i = 1, 2, \ldots, m \tag{2}$$

$$h_i(x) = 0 \qquad i = 1, 2, \ldots, p \tag{3}$$

where k is the number of objective functions $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$, $g_i, h_j : \mathbb{R}^n \rightarrow \mathbb{R}, i = 1, ..., m, j = 1, ..., p$ are the constraint functions of the problem and $x = [x_1, x_2, \ldots, x_n]^T$ the vector of decision variables. We thus wish to determine from the set Ω (where Ω is the feasible region) of all the vectors that satisfy (2) and (3) to the vector $x^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ of solutions that are *Pareto optimal*. To describe the concept of optimality that we will adopt, we need to introduce a few additional definitions.

Pareto Optimality: We say that a vector of decision variables $x^* \in \Omega$ is **Pareto Optimal** with respect to Ω if and only if $\forall x \in \Omega \wedge \forall i \in \{1, \ldots, k\}$:

$$f_i(x) = f_i(x^*) \vee \nexists i \in \{1, \ldots, k\} : f_i(x) < f_i(x^*) \tag{4}$$

Pareto Dominance: A vector $u = [u_1, \ldots, u_k]^T$ is said to **dominate** another vector $v = [v_1, \ldots, v_k]^T$ (denoted by $u \preceq v$) if and only if u is partially less than v, i.e.,:

$$\forall i \in \{1, \ldots, k\}, u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i \tag{5}$$

Pareto Optimal Set: For a given MOP $f(x)$, the **Pareto Optimal Set** P^* is defined by:

$$P^* := \{x \in \Omega \mid \nexists x' \in \Omega, f(x') \preceq f(x)\} \tag{6}$$

Pareto Front: For a given MOP $f(x)$ and its Pareto optimal set P^*, the **Pareto Front** PF^* is defined by:

$$PF^* := \{f(x) \mid x \in P^*\} \tag{7}$$

When plotted in objective space, the nondominated vectors are collectively known as the **Pareto front**.

III. Previous Related Work

Regarding studies on parameter scalability in MOEAs, the most significant ones that we are aware of are those reported by Durillo et al. [6], [7], in which the behavior and effect of parameter scalability over eight state-of-the-art multi-objective metaheuristics is analyzed. Such metaheuristics include three genetic algorithms (GAs) (NSGA-II, SPEA2 and PESA-II), an evolution strategy (PAES), a PSO algorithm (OMOPSO), a cellular GA (MOCell), an algorithm based on differential evolution (GDE3) and a Scatter Search algorithm (AbYSS). All of these approaches were studied when solving a benchmark of parameter-wise scalable problems (the ZDT [3] test suite). The authors analyzed the behavior of these eight multi-objective metaheuristics when adopting a number of decision variables that ranged from 8 up to 2048. The hypervolume performance indicator [11] was adopted to define a stopping criterion. The study paid particular attention to the computational effort required by each algorithm for reaching the true Pareto front of each problem. These papers provide empirical evidence of the decrease in efficacy and efficiency that multi-objective metaheuristics have when dealing with MOPs with a large number of decision variables, as it is shown in their results.

Another work in this direction is a small study presented in [12], where ZDT1 is solved with up to 100 decision variables using MOEA/D. They analyzed how the computational cost of their approach, in terms of the number of function evaluations, increased as the number of decision variable of the problem increased. This was shown using a number of decision variables that ranged from 10 up to 100 variables. They adopted as a performance index the average number of function evaluations used by MOEA/D for reducing the D-metric [13] and concluded that the average number of function evaluations scales up linearly, as the number of decision variables increases. They attribute these results to two facts: (1) the number of scalar optimization sub-problems in MOEA/D is fixed to be 100, regardless of the number of decision variables of the problem, and (2) the complexity of each scalar optimization could scale linearly with the number of decision variables.

However, this study is too small to show a general behavior of MOEA/D over large scale (in decision variables space) MOPs. Later on, an algorithm based on interdependence variable analysis and control variable analysis designed to deal with large scale MOPs was presented in [14]. This approach, called MOEA/DVA, decomposes a MOP with high dimensionality into a set of simpler sub-MOPs with low-dimensional subcomponents. Based on interdependent analysis between two variables, decision variables are decomposed into several low-dimensional subcomponents. Each sub-MOP independently optimizes subcomponents one by one. This work was then improved in [15], where the decomposition is based on a decision variable clustering method. Following the idea of MOEA/DVA, the proposed approach, called LMEA, uses a decision variable clustering method to divide the decision variables into convergence-related and diversity-related ones.

Different from MOEA/DVA, instead of adopting a decision variable analysis method based on dominance relationships, LMEA adopts a decision variable clustering method based on the k-means method with features measured by the angles between the sample solutions and the direction of convergence, where smaller angles indicate more contributions to convergence and larger angles indicate more contributions to diversity.

Large-scale optimization has been the focus of an important amount of research in global (single-objective) optimization using evolutionary algorithms. From these methods, cooperative coevolution has been found to be one of the most successful approaches for solving large and complex problems, through the use of problem decomposition. There is plenty of evidence of the success of this sort of approach in large scale global optimization [9], [10], [16], so we adopt here some of these ideas to develop our proposal.

IV. Our Proposed Approach

The main idea of our proposed approach is to make use of the divide-and-conquer technique, adopted by the cooperative coevolutionary framework for large scale single objective optimization. Here, we propose a new way of applying coevolutionary collaborations in terms of crossover operations which makes posible for a regular MOEA to improve its performance in large scale MOPs. Next, we provide a brief description of cooperative coevolution.

A. Cooperative coevolution

In nature, coevolution is the process of reciprocal genetic change in one species, or group, in response to another. That is, coevolution refers to a reciprocal evolutionary change between species that interact with each other [17]. A coevolutionary search involves the use of multiple species as the representation of a solution to an optimization problem. Each species can either compete or cooperate during the evolutionary process. Therefore, such models have been historically categorized as *competitive* or *cooperative*. In the case of cooperative algorithms, which are the focus of this work, individuals are rewarded when they work well with other individuals and punished when they perform poorly together [18]. Each population represents a piece of a larger problem, and it is the task of those populations to evolve increasingly fit pieces for the larger problem.

The first framework of cooperative coevolution (CC) utilized within evolutionary algorithms was originally introduced by Potter and De Jong [19], with their *Cooperative Coevolutionary Genetic Algorithm* (CCGA). This framework uses a divide-and-conquer approach to split the decision variables into subpopulations of smaller size, so that each of these subpopulations is optimized with a separate EA. The main idea was to decompose a *high-dimensional* problem into several low-dimensional subcomponents and evolve these subcomponents cooperatively. So, instead of evolving a population (globally or spatially distributed) of similar individuals representing a global solution, the cooperative coevolutionary framework co-evolves subpopulations of individuals representing specific parts of the global solution.

After this work, there were many more *cooperative coevolutionary* approaches, most of which were adopted to solve large scale global optimization problems [9], [10], [16], [20]. In general, the most common cooperative coevolutionary framework for high-dimensional global (single-objective) optimization can be summarized as follows:

1) Decompose an objective vector into m low-dimensional subcomponents.
2) Set $j = 1$ to start a new cycle.
3) Optimize the j^{th} subcomponent with a certain EA for a pre-defined number of fitness evaluations (FEs).
4) If $j < m$ then $j++$, and go to Step 3.
5) Finish if the stopping criteria are satisfied; otherwise, go to Step 2 for the next cycle.

Here, a cycle consists of one complete evolution of all species using certain EA. The advantage of cooperative coevolutionary algorithms is the decomposition of the problem which allows us to learn different parts of the problem instead of the whole problem at once.

B. Description of our proposed approach

One of the main drawbacks in the use of cooperative coevolution is the adoption of multiple subpopulations. Since each of these populations makes use of independent function evaluations, this makes a cooperative coevolutionary algorithm (CCA) to be more expensive (in terms of function evaluations) than a regular MOEA. If we are to extend the basic computational model of cooperative coevolution into an approach that does not make use of extra function evaluations, we cannot adopt the whole model of cooperative coevolution, since it would be much more costly than the use of a MOEA as a standalone algorithm. In other words, we must find a way to use the key concept of cooperative coevolution (problem decomposition) without the need of having multiple subpopulations. For this sake, we propose here the so-called *operational decomposition* approach, which is a coevolutionary step added to a MOEA, where we make use of the divide-and-conquer technique that splits the MOP to be solved (in *decision variables* space) when applying crossover.

The first concept we must address is problem decomposition in decision variable space. Problem decomposition consists in determining an appropriate number of subcomponents and the role that each of them will play in the overall search process. For some problems, an appropriate form of decomposition may be known *a priori* but for others, this may not be possible. Let's consider the problem of optimizing a function of m independent variables. It may be reasonable to decompose the problem into m subtasks, with each of them being assigned to the optimization of a single variable. However, there are many problems for which we have little or no information related to the number or roles of subcomponents that should be used in the decomposition. This occurs in non-separable functions. Here, non-separable means that the vector of decision variables is composed by elements that interact with each other and are

not independent. There is evidence that indicates that dividing the problem into random groups provides better results than applying a deterministic division scheme, when dealing with non-separable functions [9], [16]. Following this idea, our proposed approach divides the vector of decision variables into S subcomponents (species), each of them representing a subset of all the decision variables at a time rather than taking only one variable per subcomponent. We assign each decision variable to its correspondent subcomponent in a random way, trying to increase the chance of optimizing some interacting variables together.

Our proposed decomposition approach is an adaptation of cooperative coevolution (CC) applied to the crossover and mutation operators that does not need individuals from the other species to assemble a complete solution in order to perform a fitness evaluation. Instead, it uses other individuals for the creation of new solutions. We only use decision variable decomposition to perform crossover operations which allows us to handle in a better way the *curse of dimensionality* (the performance of an evolutionary algorithm deteriorates rapidly as the dimensionality of the search space increases [21]) present in MOEAs. So, individuals will still be representing a whole solution, but operators will be applied based on the corresponding species, and not based on the individuals. This makes the crossover operator to be more effective, since decomposition of the operations causes a bigger effect than when adopting the usual scheme in which most MOEAs are implemented. The algorithm of our proposed decomposition scheme, when incorporated to a MOEA, works as follows:

Input:

- The MOP (1)
- S: The number of species for decision variables decomposition
- T: The neighborhood size to apply coevolutionary collaboration

Output:

- PS: the final solutions found during the search

Step 1) **Initialization**:

Step 1.1) Set the external population of final solutions $PS = \emptyset$.

Step 1.2) Generate an initial population $X = x^1, \dots x^N$ randomly or by a problem-specific method. Set $FV^i = f(x^i)$.

Step 1.3) Divide the problem into S subcomponents $c^1, \dots, c^S$ each one of dimension m, created in a random way from the original vector of decision variables x of dimension D (as shown in Figure 1), where $D = m * S$, such that, for each $j = 1, \dots, N$, $x^j = [c^1_j, \dots, c^S_j]$.

Step 2) **Update**:

Step 2.1) Find the T closest decision variables vectors to each solution $x_i \in X$. For each x_i, set $B(i) = \{i_1, \dots, i_T\}$, where $x^{i_1}, \dots, x^{i_T}$ are the T closest solutions to x^i .

For $i = 1, \dots, N$ do

Step 2.2) **Decompositional Crossover Operation and Mutation:**

For $j = 1, \dots, S$ do

Step 2.2.1) Randomly select two indexes p, q from $B(i)$, and then generate a new solution y^j_c from c^j_p and c^j_q using crossover.

Step 2.2.2) Apply a problem-specific crossover operation on y^j_c to produce ${y'}^j_c$.

Step 2.3) Assemble y' from $[{y'}^1_c, \dots, {y'}^S_c]$, sorting the subcomponents to form the original vector of decision variables.

Step 2.4) Apply mutation operator and evaluate solution y'.

Step 2.5) Remove from the external population PS all the vectors dominated by $f(y')$. Add $f(y')$ to PS if no vectors in PS dominate it.

Step 3) **Stopping Criterion**: Stop if the termination criterion is satisfied. Otherwise, go to Step 2.

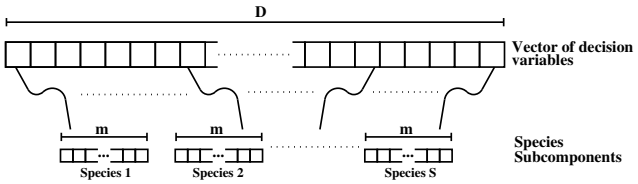

Fig. 1. Graphical representation of the subcomponents (species) creation. Here, we assume a vector of decision variables of dimension D which is divided into S subcomponents of dimension m, created in a random way from the original vector of decision variables and assigned to the S existing species, where $D = m * S$.

Since c^j_p and c^j_q in Step 2.2.1 are subcomponent (in decision variables space) neighbor solutions and their dimensionality is lower than that of the original vector of decision variables x, their offspring ${y'}^j_c$ (later improved by mutation) should be a good contribution to the complete assemble of the new final solution y'. The use of neighbors allow the new solution to have a more controlled modification and the absence of it makes the approach to have a very poor performance, causing a poor convergence of the solutions. By using only the decomposition nature of the cooperative coevolutionary framework into the MOEA's crossover operation, there is no need for extra function evaluations. Therefore, the efficiency of the adopted MOEA is not lost.

C. Experimental Results

In order to validate our approach we adopted NSGA-III [8] and incorporated our decomposition approach to it, giving rise to a novel MOEA called OD-NSGA. NSGA-III is the third version of the Non-dominated Sorting Genetic Algorithm (NSGA), which is intended for many-objective optimization.

In fact, NSGA-III is an extension of the NSGA-II [22] in which the crowding comparison operator is replaced by a clustering operator (aided by a set of well-distributed reference points). This approach modifies the selection mechanism of NSGA-II by performing an analysis of the distances of the individuals in the population with respect to the supplied reference points, preferring population members that are non-dominated and close to such reference points. We validated OD-NSGA comparing its performance with respect to that of the original NSGA-III.

1) Methodology: For the purposes of this study, we adopted the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [4] and the Walking Fish Group test suite [5] with instances of three objectives with a number of decision variables that ranges from 200 to 1200. In order to assess the performance of each approach, we selected the hypervolume indicator [11], since this measure can differentiate between degrees of complete outperformance of two sets. The hypervolume is defined as the n-dimensional space that is contained by an n-dimensional set of points. When applied to multi-objective optimization, the n-dimensional objective values for solutions are treated as points for the computation of such space. That is, the hypervolume is obtained by computing the volume (in objective function space) of the nondominated set of solutions Q that minimize a MOP. For every solution $i \in Q$, a hypercube v_i is generated with a reference point W and the solution i as the diagonal corner of the hypercube:

$$\mathcal{S} = Vol\left(\bigcup_{i=1}^{|Q|} v_i\right) \tag{8}$$

The aim of this study is to identify which of the algorithms being compared is able to get closer to the true Pareto front using the same number of objective function evaluations and how they behave as the dimensionality of the MOP increases.

D. Parameterization

The parameters of each algorithm adopted in our study were chosen in such a way that we could do a fair comparison among them. For both OD-NSGA and NSGA-III, we adopted Simulated-binary crossover (SBX) and polynomial-based mutation [22] as the crossover and mutation operators, respectively. The mutation probability was set to $p_m = 1/l$, where l is the number of decision variables; the distribution indexes for SBX and polynomial-based mutation were set as: $\eta_c = 20$ and $\eta_m = 20$. For the case of OD-NSGA, different numbers of species were used for each problem instance, in order to have 2 decision variables per species. So, for all problems a number $S = l/2$ of species was used. The maximum number of iterations adopted for all problems and both MOEAs was set to 1000, regardless of their dimensionality. Finally, the population size for both algorithms in all problem instances was set to 120 and the number of supplied reference points was set to 12.

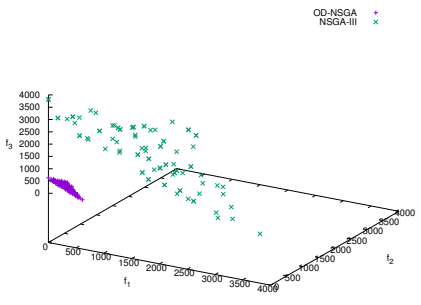

Fig. 2. Plot of DTLZ1 with 200 decision variables.

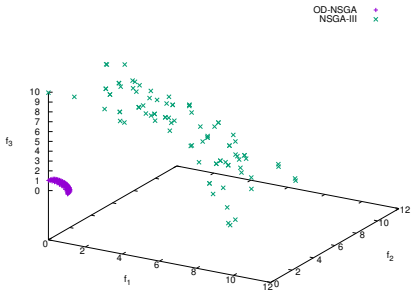

Fig. 3. Plot of DTLZ2 with 200 decision variables.

E. Discussion of Results

In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Tables I and II show the average hypervolume value of each of the two MOEAs being compared for each test problem adopted, as well as the results of the statistical analysis that we made to validate our experiments, for which we adopted Wilcoxon's rank sum. The cells containing the best hypervolume value for each problem have a grey colored background. Also, we show the improvement on the hypervolume value that our approach was able to obtain against the original NSGA-III. From Figures 2 to 12, we plotted the results of the median of the 25 runs for OD-NSGA and NSGA-III for the adopted test problems with 200 decision variables.

NSGA-III produced competitive results for the WFG test problems, although it could not outperform our approach in any problem instance. According to Wilcoxon's test, we cannot reject the null hypothesis in any case when comparing our approach to NSGA-III. Regarding the DTLZ test problems, our approach outperformed NSGA-III in a more remarkable way, and as the results show, as the dimensionality of the problems grows, the improvement obtained by our approach on the hypervolume value increases. Based on the results of Wilcoxon's test, we can confirm that the null hypothesis can be rejected, so OD-NSGA yields the best overall results. Also, from Figures 2 to 8 we can observe that, using the same number of function evaluations, OD-NSGA is able to get closer than NSGA-III to the true Pareto front in all problems.

Function	No. Vars	NSGAIII HV	ODNSGA HV	ODNSGA - NSGAIII Improvement	P(H)
DTLZ1	200	215977135388000	216000997005000	23861617000	0.000000 (1)
	400	215400012903000	215996994093000	596981190000	0.000000 (1)
	600	212974387627000	215950892918000	2976505291000	0.000000 (1)
	800	207540202162000	215621649556000	8081447394000	0.000000 (1)
	1000	196665040413000	214698409969000	18033369556000	0.000000 (1)
	1200	181783052866000	211861891166000	30078838300000	0.000000 (1)
DTLZ2	200	1002373	1003002	629	0.000000 (1)
	400	990803	1003002	12198	0.000000 (1)
	600	949906	1002996	53090	0.000000 (1)
	800	866389	1002945	136556	0.000000 (1)
	1000	729303	1002714	273411	0.000000 (1)
	1200	523570	1002097	478528	0.000000 (1)
DTLZ3	200	1727842691130000	1728003283920000	160592790000	0.000000 (1)
	400	1719429195360000	1727973502150000	8544306790000	0.000000 (1)
	600	1675915201830000	1727710404470000	51795202640000	0.000000 (1)
	800	1573081147930000	1726309252600000	153228104670000	0.000000 (1)
	1000	1375302245890000	1719176246810000	343874000920000	0.000000 (1)
	1200	1064140471370000	1696259566570000	632119095200000	0.000000 (1)
DTLZ4	200	1001798	1002998	1201	0.000000 (1)
	400	972337	1002467	30130	0.000000 (1)
	600	900392	1002411	102019	0.000000 (1)
	800	821398	1002887	181489	0.000000 (1)
	1000	697440	1002501	305061	0.000000 (1)
	1200	493776	1000888	507112	0.000000 (1)
DTLZ5	200	1000999	1002869	1870	0.000000 (1)
	400	986798	1002847	16049	0.000000 (1)
	600	945228	1002667	57439	0.000000 (1)
	800	865392	1002238	136847	0.000000 (1)
	1000	729668	1001604	271936	0.000000 (1)
	1200	524690	1000446	475756	0.000000 (1)
DTLZ6	200	1724407391	1727851741	3444350	0.000000 (1)
	400	1693018891	1720210682	27191791	0.000000 (1)
	600	1611572531	1691995048	80422517	0.000000 (1)
	800	1468395277	1626832244	158436967	0.000000 (1)
	1000	1236630246	1509098688	272468442	0.000000 (1)
	1200	913582585	1318241856	404659271	0.000000 (1)
DTLZ7	200	3033	6834	3801	0.000000 (1)
	400	2346	6128	3782	0.000000 (1)
	600	2065	5578	3513	0.000000 (1)
	800	1955	5226	3271	0.000000 (1)
	1000	1887	4934	3047	0.000000 (1)
	1200	1825	4707	2882	0.000000 (1)

TABLE I

AVERAGE OF THE HYPERVOLUME INDICATOR VALUES OF THE RESULTS OBTAINED FOR THE DTLZ TEST PROBLEMS. WE SHOW AVERAGE VALUES OVER 25 INDEPENDENT RUNS. THE CELLS CONTAINING THE BEST HYPERVOLUME VALUE FOR EACH PROBLEM HAVE A GREY COLORED BACKGROUND. THE IMPROVEMENT COLUMNS SHOW THE IMPROVEMENT ON THE HYPERVOLUME VALUE THAT OUR APPROACH WAS ABLE TO GET AGAINST THE OTHER ALGORITHM. THE P(H) COLUMNS SHOWS THE RESULTS OF THE STATISTICAL ANALYSIS APPLIED TO OUR EXPERIMENTS USING WILCOXON'S RANK SUM. P IS THE PROBABILITY OF OBSERVING THE GIVEN RESULT (THE NULL HYPOTHESIS IS TRUE). SMALL VALUES OF P CAST DOUBT ON THE VALIDITY OF THE NULL HYPOTHESIS. $H = 0$ INDICATES THAT THE NULL HYPOTHESIS ("MEDIANS ARE EQUAL") CANNOT BE REJECTED AT THE 5% LEVEL. $H = 1$ INDICATES THAT THE NULL HYPOTHESIS CAN BE REJECTED AT THE 5% LEVEL.

Function	No. Vars	NSGAIII HV	ODNSGA HV	ODNSGA - NSGAIII Improvement	P(H)
WFG1	200	19	27	7	0.000000 (1)
	400	17	26	9	0.000000 (1)
	600	16	25	9	0.000000 (1)
	800	16	25	10	0.000000 (1)
	1000	15	25	10	0.000000 (1)
	1200	15	25	10	0.000000 (1)
WFG2	200	256	323	67	0.000000 (1)
	400	243	305	62	0.000000 (1)
	600	241	304	63	0.000000 (1)
	800	238	308	70	0.000000 (1)
	1000	237	292	55	0.000000 (1)
	1200	233	301	68	0.000000 (1)
WFG3	200	253	278	24	0.000000 (1)
	400	242	272	29	0.000000 (1)
	600	239	268	29	0.000000 (1)
	800	236	266	29	0.000000 (1)
	1000	234	264	30	0.000000 (1)
	1200	232	262	30	0.000000 (1)
WFG4	200	274	317	43	0.000000 (1)
	400	245	311	66	0.000000 (1)
	600	228	308	80	0.000000 (1)
	800	219	305	87	0.000000 (1)
	1000	208	303	95	0.000000 (1)
	1200	210	301	91	0.000000 (1)
WFG5	200	261	312	51	0.000000 (1)
	400	243	304	62	0.000000 (1)
	600	232	298	66	0.000000 (1)
	800	226	294	68	0.000000 (1)
	1000	225	291	66	0.000000 (1)
	1200	215	288	73	0.000000 (1)
WFG6	200	255	317	62	0.000000 (1)
	400	234	308	73	0.000000 (1)
	600	220	300	79	0.000000 (1)
	800	213	294	81	0.000000 (1)
	1000	209	289	80	0.000000 (1)
	1200	207	285	78	0.000000 (1)
WFG7	200	246	317	71	0.000000 (1)
	400	228	309	81	0.000000 (1)
	600	220	302	81	0.000000 (1)
	800	214	297	83	0.000000 (1)
	1000	212	293	81	0.000000 (1)
	1200	213	291	78	0.000000 (1)
WFG8	200	251	311	61	0.000000 (1)
	400	227	306	79	0.000000 (1)
	600	223	303	80	0.000000 (1)
	800	217	300	83	0.000000 (1)
	1000	213	298	85	0.000000 (1)
	1200	214	296	81	0.000000 (1)
WFG9	200	233	296	62	0.000000 (1)
	400	224	285	61	0.000000 (1)
	600	220	277	57	0.000000 (1)
	800	218	273	56	0.000000 (1)
	1000	215	270	55	0.000000 (1)
	1200	213	267	53	0.000000 (1)

TABLE II

AVERAGE OF THE HYPERVOLUME INDICATOR VALUES OF THE RESULTS OBTAINED FOR THE WFG TEST PROBLEMS.

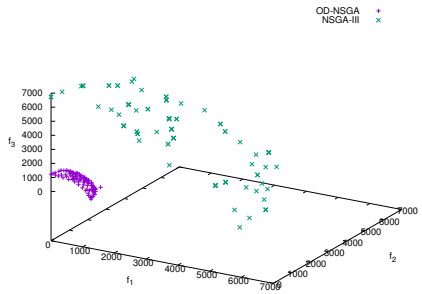

Fig. 4. Plot of DTLZ3 with 200 decision variables.

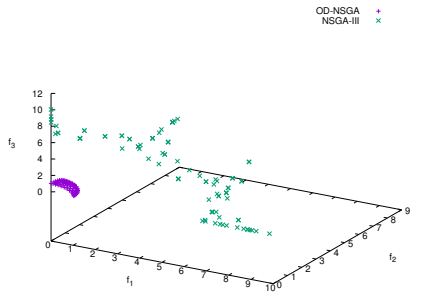

Fig. 5. Plot of DTLZ4 with 200 decision variables.

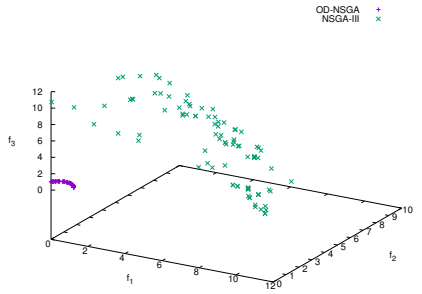

Fig. 6. Plot of DTLZ5 with 200 decision variables.

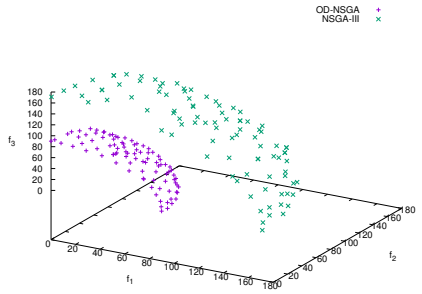

Fig. 7. Plot of DTLZ6 with 200 decision variables.

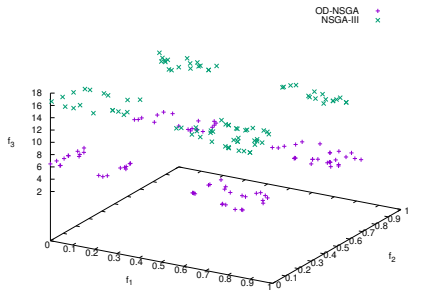

Fig. 8. Plot of DTLZ7 with 200 decision variables.

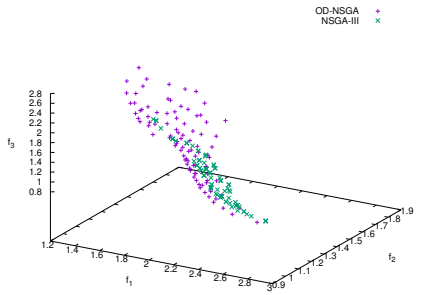

Fig. 9. Plot of WFG1 with 200 decision variables.

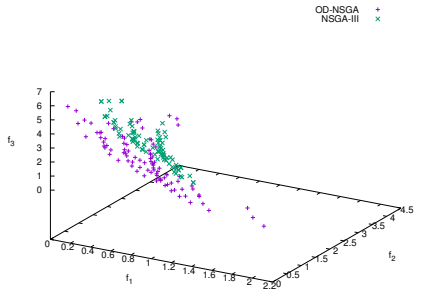

Fig. 10. Plot of WFG2 with 200 decision variables.

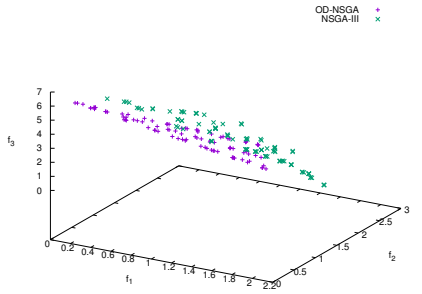

Fig. 11. Plot of WFG3 with 200 decision variables.

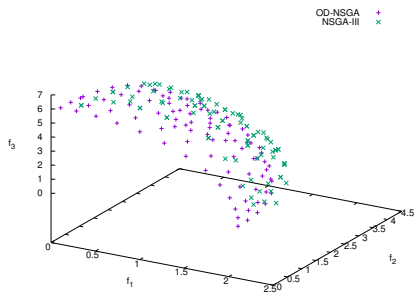

Fig. 12. Plot of WFG4 with 200 decision variables.

V. Conclusions and Future Work

Here, we developed a novel MOEA based on our coevolutionary like operators, which adopts decomposition based techniques used by cooperative coevolutionary algorithms. Our approach, called OD-NSGA, is based on a new scheme that applies the crossover operator in a coevolutionary manner, which make it capable of solving problems with many decision variables. Our experimental results indicate that when adopting operational decomposition, NSGA-III was able to improve its performance in MOPs having from 200 up to 1200 decision variables. Our approach was able to deal with all the difficulties presented in the DTLZ and WFG test suites, even in high dimensionality. The results confirmed that our proposed approach is very effective and efficient in tackling large scale MOPs. As part of our future work, we intend to study other decomposition techniques for decision variable space. Also, we want to test our approach in many-objective MOPs. We are also interested in studying the incorporation of operational decomposition in other state-of-the-art MOEAs.

Acknowledgment

The second author gratefully acknowledges support from CONACyT project no. 1920 and from a 2018 SEP-Cinvestav grant (application no. 4).

References

[1] R. C. Walker, *Introduction to Mathematical Programming*. USA: Pearson Learning Solution, 2012.

[2] C. A. Coello Coello, D. A. Van Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*. New York: Kluwer Academic Publishers, May 2002, iSBN 0-3064-6762-3.

[3] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, Tech. Rep. 70, December 1999.

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

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

[6] J. J. Durillo, A. J. Nebro, C. A. Coello Coello, F. Luna, and E. Alba, "A Comparative Study of the Effect of Parameter Scalability in Multi-Objective Metaheuristics," in *2008 Congress on Evolutionary Computation (CEC'2008)*. Hong Kong: IEEE Service Center, June 2008, pp. 1893–1900.

[7] J. Durillo, A. Nebro, C. Coello Coello, J. Garcia-Nieto, F. Luna, and E. Alba, "A Study of Multiobjective Metaheuristics When Solving Parameter Scalable Problems," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 4, pp. 618–635, August 2010.

[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*, vol. 18, no. 4, pp. 577–601, August 2014.

[9] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in *2010 IEEE Congress on Evolutionary Computation (CEC'2010)*. Barcelona, Spain: IEEE Press, July 2010, pp. 1–8.

[10] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, August 2008.

[11] E. Zitzler, D. Brockhoff, and L. Thiele, "The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicator Via Weighted Integration," in *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds. Matshushima, Japan: Springer. Lecture Notes in Computer Science Vol. 4403, March 2007, pp. 862–876.

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

[13] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert da Fonseca, "Performance Assessment of Multiobjective Optimizers: An Analysis and Review," Computer Engineering and Networks Laboratory, ETH Zurich, Tech. Rep. 139, June 2002.

[14] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, April 2016.

[15] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, 2018.

[16] Z. Yang, K. Tang, and X. Yao, "Differential Evolution for High-Dimensional Function Optimization," in *2007 IEEE Congress on Evolutionary Computation*. Singapore: IEEE Press, September 2007, pp. 3523–3530, iSBN 978-1-4244-1339-3.

[17] P. R. Ehrlich and P. H. Raven, "Butterflies and Plants: A Study in Coevolution," *Evolution*, vol. 18, no. 4, pp. 586–608, 1964.

[18] M. A. Potter and K. A. D. Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evolutionary Computation*, vol. 8, no. 1, pp. 1–29, March 2000.

[19] ——, "A Cooperative Coevolutionary Approach to Function Optimization," in *Parallel Problem Solving from Nature–PPSN III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. Jerusalem, Israel: Springer-Verlag, October 1994, pp. 249–257.

[20] Z. Yang, K. Tang, and X. Yao, "Multilevel Cooperative Coevolution for Large Scale Optimization," in *2008 IEEE Congress on Evolutionary Computation (CEC'2018)*. Hong Kong: IEEE Press, 2008, pp. 1663–1670.

[21] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, June 2004.

[22] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II," Indian Institute of Technology, Kanpur, India, KanGAL report 200001, 2000.