Decomposition-based Approach for Solving Large Scale Multi-objective Problems

Luis Miguel Antonio and Carlos A. Coello Coello*

CINVESTAV-IPN (Evolutionary Computation Group)
Departamento de Computación
México D.F. 07300, MÉXICO
lmiguel@computacion.cs.cinvestav.mx, ccoello@cs.cinvestav.mx

Abstract. Decomposition is a well-established mathematical programming technique for dealing with multi-objective optimization problems (MOPs), which has been found to be efficient and effective when coupled to evolutionary algorithms, as evidenced by MOEA/D. MOEA/D decomposes a MOP into several single-objective subproblems by means of well-defined scalarizing functions. It has been shown that MOEA/D is able to generate a set of evenly distributed solutions for different multi-objective benchmark functions with a relatively low number of decision variables (usually no more than 30). In this work, we study the effect of scalability in MOEA/D and show how its efficacy decreases as the number of decision variables of the MOP increases. Also, we investigate the improvements that MOEA/D can achieve when combined with co-evolutionary techniques, giving rise to a novel MOEA which decomposes the MOP both in objective and in decision variables space. This new algorithm is capable of optimizing large scale MOPs and outperforms MOEA/D and GDE3 when solving problems with a large number of decision variables (from 200 up to 1200).

1 Introduction

Although in real-world applications, many MOPs have hundreds or even thousands of decision variables, the effect of the scalability of decision variables space over modern MOEAs has not been properly addressed. In fact, scalability in decision variables space is a topic that has been only scarcely studied in the context of multi-objective optimization using 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 [5, 4]. The work reported here tries to narrow the gap in this important topic.

* The second author gratefully acknowledges support from CONACyT project no. 221551.

We are interested in improving the MOEA/D [14] framework in order to make it capable to deal with large scale (in decision variables space) MOPs. Thus, we study here the effect of scalability in MOEA/D and we investigate the improvements that this algorithm can achieve. For this purpose, we propose to combine the MOEA/D framework with Cooperative Coevolutionary techniques (which have shown to be very effective for large scale single-objective optimization [8, 13]), giving rise to a novel MOEA based on a double decomposition (both in objective and decision variables space).

The remainder of this paper is organized as follows. The previous related work is discussed in Section 2. Section 3 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 4.

2 Previous Related Work

Regarding studies on scalability in MOEAs, to the authors' best knowledge, the most significant ones are those reported by Durillo et al. [5, 4], in which the behavior and effect of decision variables scalability over eight multi-objective metaheuristics (representatives of the state-of-the-art) are analyzed. For this sake, the authors adopted a benchmark of scalable problems (the Zitzler-Deb-Thiele (ZDT) [16] test suite) using a number of decision variables that ranged from 8 up to 2048. 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 [14], where ZDT1 is solved with up to 100 decision variables using MOEA/D. They analyze how the computational cost, measured in terms of the number of function evaluations, increases as the number of decision variables of the problem increases. This is shown using a number of decision variables that ranges from 10 up to 100 variables. They used as a performance index the average number of function evaluations spent by MOEA/D for reducing the D-metric [17] and concluded that the average number of function evaluations linearly scales up, as the number of decision variables increases. They attribute these results to two facts: i) 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. ii) the complexity of each single-objective 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.

Although scalability in decision variables space is a topic that has been only scarcely studied in the evolutionary multi-objective optimization field, large-scale optimization has been the focus of an important amount of research in global (single-objective) optimization using evolutionary algorithms. The currently available approaches for large-scale global optimization can be roughly

divided in two groups: those that decompose a high-dimensional decision variables vector into small subcomponents which can then be handled by conventional EAs (see for example [13]) and the ones that approach the problem by disturbing the population of the EA or by combining different evolutionary methods (see for example [9]). 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.

3 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, and incorporate such concept into MOEA/D. Our motivation is that it is very natural to use scalar optimization methods in MOEA/D, since each solution is associated with a scalar optimization problem, in contrast with non-decomposition MOEAs where in most cases there is no easy way for them to take advantage of scalar optimization methods. Next, we give a brief description of both MOEA/D and cooperative coevolution.

3.1 MOEA/D

The *multi-objective evolutionary algorithm based on decomposition* (MOEA/D) [14] has attracted growing interest from the community, due to its simplicity and to its effectiveness when applied to a broad range of MOPs. MOEA/D decomposes the MOP into a set of single-objective subproblems and solves these subproblems simultaneously using an evolutionary algorithm. It adopts a set of weights each of which corresponds to a single subproblem. Each weight vector is used as a search direction to define a scalar function. For this sake, the so called Tchebycheff decomposition is the most widely used approach. Given a weight vector $\lambda = [\lambda_1, \ldots, \lambda_n]^T$ the corresponding subproblem is defined as:

$$\text{minimize} \quad g^{te}(x|\lambda, z^*) = \max_{1 \leq i \leq n} \lambda_i |f_i(x) - z_i^*| \tag{1}$$

where z^* is the reference point chosen as the minimum of objective function values found during the evolution. The main advantage of the Tchebycheff approach is that it works regardless of the shape of the Pareto front, while other decomposition approaches (like the weighted sum approach) only work for convex Pareto fronts. The weights are also used to define neighborhoods of the subproblems. The neighborhood relations among these subproblems are defined based on the distances between their aggregation coefficient vectors. At each generation, a new individual is generated and evaluated using its own neighborhood of weights, with the idea that any information about these closest weight vectors should be helpful for optimizing the current individual's subproblem. Once this new individual is created, it is compared to its parent and in case it is

better, it replaces its parent. Moreover, it is also compared to other individuals in its neighborhood and is allowed to replace some of them. Therefore, at each generation, the population is composed of the best solution found so far (i.e., since the start of the run of the algorithm) for each subproblem.

3.2 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 [6]. A coevolutionary search involves the use of multiple species as the representation of a solution to an optimization problem. 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 [11].

The first framework of cooperative coevolution (CC) utilized within evolutionary algorithms was originally introduced by Potter and De Jong [10], 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 (global or spatially distributed) of similar individuals representing a global solution, the cooperative coevolutionary framework coevolves subpopulations of individuals representing specific parts of the global solution.

After this work, there were many more *cooperative coevolutionary* approaches, most of them for large scale global optimization since this showed to be a good framework for solving high-dimensional problems [8, 13]. In general, the most common cooperative coevolutionary framework for high-dimensional global (single-objective) optimization can be summarized as follows:

1. Decompose a vector of decision variables 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 predefined number of fitness evaluations (FEs).
4. If $j < m$ then $j++$, and go to Step 3.
5. Stop if the stopping criteria are satisfied; otherwise, go to Step 2 for the next cycle.

3.3 Description of our proposed approach

If we are to extend the basic computational model of cooperative coevolution into an approach that already uses a decomposition strategy as the one adopted by MOEA/D, we must address the issues of a second problem decomposition, as well as other issues such as the interdependencies among subcomponents, credit

assignment, and the maintenance of diversity. In order to do so and to provide reasonable opportunities for the success of co-adapted subcomponents and an increase in efficiency when dealing with large scale MOPs, we can not use the whole model of cooperative coevolution as we did in our previous work presented in [1], since it is much more costly (due to the use of multiple subpopulations) than the use of MOEA/D as a standalone algorithm. Instead, we only incorporate into MOEA/D a coevolutionary step where we make use of the divide-and-conquer technique that splits the MOP to be solved, but in *decision variables* space.

Our proposed approach divides the vector of decision variables into S subcomponents (species), each one 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 corresponding subcomponent in a random way, trying to increase the chance of optimizing some interacting variables together. However, it is important to note that the cooperative coevolutionary adaptation presented here does not work as in the original framework, since we do not intend to use several subpopulations for each subcomponent of the problem and we will not need individuals from the other species to assemble a complete solution in order to perform a fitness evaluation. Here, we only use decision variable decomposition to make operations (crossover and mutation) more effective and with this, we can manage in a better way the *curse of dimensionality* (the performance of an evolutionary algorithm deteriorates rapidly as the dimensionality of the search space increases [12]) 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. The algorithm of our proposed MOEA based on double decomposition (MOEA/D^2) works as follows:

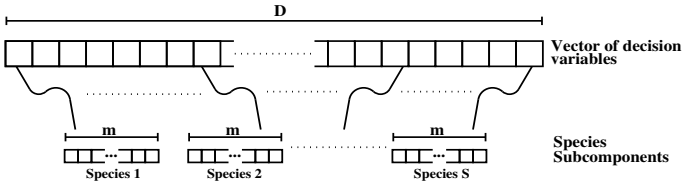

Fig. 1: Graphical representation of the subcomponents (species) creation. Here, we assume a vector of decision variables of dimension D which is divided into S subcomponents of dimension m, created in a random way from the original vector of decision variables and assigned to the S existing species,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

Input:

- The MOP with k objective functions
- N: The number of subproblems considered in MOEA/D
- S: The number of species for decision variables decomposition
- A set of N uniform spread weight vectors: $\lambda^1, \ldots, \lambda^N$
- T: The neighborhood size

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)** Find the T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$, where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i .
- **Step 1.3)** Generate an initial population $x^1, \ldots x^N$ randomly or by a problem-specific method. Set $FV^i = f(x^i)$.
- **Step 1.4)** Divide the problem into S subcomponents $c^1, \ldots, c^S$ each 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 $j = 1, \ldots, N$, $x^j = [c_j^1, \ldots, c_j^S]$.
- **Step 1.5)** Initialize $z = [z_1, \ldots, z_k]^T$, where z_i is the best value found so far for objective f_i.

Step 2) Update:
For $i = 1, \ldots, N$ do
- **Step 2.1) Crossover and Mutation:**
  For $j = 1, \ldots, S$ do
  - **Step 2.1.1)** Randomly select two indexes p, q from $B(i)$, and then generate a new solution y_c^j from c_p^j and c_q^j using crossover.
  - **Step 2.1.2)** Apply a problem-specific repair improvement heuristic on y_c^j to produce ${y'}_c^j$.
- **Step 2.2)** Assemble y' from $[{y'}_c^1, \ldots, {y'}_c^S]$, sorting the subcomponents to form the original vector of decision variables.
- **Step 2.3)** For each $j = 1, \ldots, k$, if $z_j > f_j(y')$, then set $z_j = f_j(y')$.
- **Step 2.4) Update of Neighboring Solutions**: For each index $j \in B(i)$ use (1) such that, if $g^{te}(y'|\lambda^j, z^*) < g^{te}(x^j|\lambda^j, z^*)$, then $FV^j = f(y')$.
- **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.

Since c_p^j and c_q^j in Step 2.1.1 are the current best subcomponent (in decision variables space) solutions to neighbors of the i^{th} subproblem (in objective function space) and their dimensions are less than the original vector of decision variables x, their offspring ${y'}_c^j$ (already improved by mutation) should be a good contribution to the complete assemble of the new final solution y'. Therefore, the resultant solution is very likely to have a lower (improved) function value for the neighbors of the i^{th} subproblem. Also, by using only the decomposition nature of the cooperative coevolutionary framework, there is no need for extra function evaluations. Therefore, the efficiency of MOEA/D is not lost.

3.4 Experimental Results

We validated MOEA/D^2 comparing its performance with respect to that of the original MOEA/D and with respect to GDE3 [7]. Although GDE3 is not a decomposition based algorithm, in the studies presented in [4] this differential evolution based MOEA obtained the best overall results, which is the reason why we decided to include it in our comparative study.

Methodology For the purposes of this study, we adopted the Deb-Thiele-Laumanns-Zitzler (DTLZ) test suite [3] 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 [15], 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 its diagonal corner of the hypercube:

$$\mathcal{S} = Vol\left(\bigcup_{i=1}^{|Q|} v_i\right) \tag{2}$$

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.

Parameterization The parameters of each algorithm used in our study were chosen in such a way that we could do a fair comparison among them. For MOEA/D^2 and MOEA/D, we adopted SBX and polynomial-based mutation [2] 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 the polynomial-based mutation were set as: $\eta_c = 20$ and $\eta_m = 20$. For the case of MOEA/D^2, different numbers of species were used for each problem instance, in order to have 2 decision variables per species. So, for problems with 200 decision variables, 100 species were used, for problems with 400 decision variables, 200 species were used, and so on. The maximum number of iterations adopted for all problems and MOEAs was set to 1000, regardless of their dimensionality. The F and CR values for GDE3 were set to 0.5. Finally, the population size for all algorithms in all problems instances was set to 100.

Discussion of Results In our experiments, we obtained the hypervolume value over the 25 independents runs performed. Table 1 shows the average hypervolume of each of the 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 used Wilcoxon's rank sum. Also, we show the improvement on the hypervolume value that our approach was able to obtain with respect to that of the other algorithms. GDE3 presented the poorest performance in all problem instances. MOEA/D produced competitive results for DTLZ2 and DTLZ4, although it could not outperform our approach in any problem instance. According to Wilcoxon's test, we cannot reject the null hypothesis in only two

cases when comparing our approach to MOEA/D, in DTLZ2 and DTLZ4 with 200 decision variables, which means that in these cases both algorithms have a similar behavior. This shows that our approach has a similar performance to MOEA/D in multi-frontal problems. The best overall performance of our approach was in DTLZ1, DTLZ3 and DTLZ6, where our approach significantly outperformed MOEA/D and GDE3, and as the results show, as the dimensionality of the problems grows, the improvement obtained by our approach on the hypervolume value increases. So, we can confirm that our approach can handle in a better way problems with degenerate Pareto optimal fronts, as is the case of DTLZ6. Decomposition is very effective when solving non-separable problems such as DTLZ1 and DTLZ3. For DTLZ5 and DTLZ7, the improvement was more remarkable as the dimensionality of the problems increased. However, our approach was also able to outperform both MOEA/D and GDE3 in all instances. Based on the results of Wilcoxon's test, we can confirm that the null hypothesis can be rejected, so MOEA/D^2 produced the best overall results.

4 Conclusions and Future Work

Here, we developed a novel decomposition-based MOEA called MOEA/D^2, which adopts decomposition based techniques used by cooperative coevolutionary algorithms. MOEA/D^2 uses a double decomposition of the MOP, one in objective functions space, as done by MOEA/D, and another one in decision variables space. Our experimental results indicate that MOEA/D^2 clearly outperforms MOEA/D and GDE3 in MOPs having from 200 up to 1200 decision variables. Our approach was able to deal with all the difficulties presented in the DTLZ test suite, 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. We are also interested in studying the possible use of other (computationally inexpensive) methods to generate a set of weight vectors more uniformly distributed for MOEA/D^2.

References

1. L. M. Antonio and C. A. Coello Coello. Use of Cooperative Coevolution for Solving Large Scale Multiobjective Optimization Problems. In *2013 IEEE Congress on Evolutionary Computation (CEC'2013)*, pages 2758–2765, Cancún, México, 20-23 June 2013. IEEE Press. ISBN 978-1-4799-0454-9.
2. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
3. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In A. Abraham, L. Jain, and R. Goldberg, editors, *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145. Springer Berlin Heidelberg, USA, 2005.

Function	No. Vars	MOEA/D^2 HV	MOEA/D HV	MOEA/D^2-MOEA/D Improvement	MOEA/D^2-MOEA/D P(H)	GDE3 HV	MOEAD^2-GDE3 Improvement	MOEAD^2-GDE3 P(H)
DTLZ1	200	124999991998953.0000	124999970289543.0000	21709410.2344	0.000000 (1)	124923656113375.0000	76335885578.2031	0.000000 (1)
	400	124999755014274.0000	124999298073533.0000	456940740.9063	0.000000 (1)	124181465622337.0000	818289391936.5000	0.000000 (1)
	600	124998478272908.0000	124996030413092.0000	2447859816.0625	0.000000 (1)	122039040800943.0000	2959437471964.5000	0.000000 (1)
	800	124995060011719.0000	124985536236730.0000	9523774988.8594	0.000000 (1)	117702084444497.0000	7292975567222.5300	0.000000 (1)
	1000	124986970597954.0000	124955356063479.0000	31614534475.2500	0.000000 (1)	110256296271387.0000	14730674326567.2000	0.000000 (1)
	1200	124970550659900.0000	124894718579624.0000	75832080276.4531	0.000000 (1)	99191612716078.9000	25778937943821.2000	0.000000 (1)
DTLZ2	200	728999.3904	728999.3862	0.0043	0.712386 (0)	728989.2937	10.0967	0.000000 (1)
	400	728999.3808	728999.3680	0.0128	0.043602 (1)	728321.7458	677.6350	0.000000 (1)
	600	728999.3605	728999.3085	0.0520	0.000000 (1)	721831.2296	7168.1309	0.000000 (1)
	800	728999.2870	728999.1289	0.1581	0.000000 (1)	698874.5807	30124.7063	0.000000 (1)
	1000	728999.0954	728998.4267	0.6687	0.000000 (1)	653788.8012	75210.2941	0.000000 (1)
	1200	728998.4393	728994.9746	3.4647	0.000000 (1)	571671.4472	157326.9921	0.000000 (1)
DTLZ3	200	1727999970755560.0000	1727999849624200.0000	121131355.7500	0.000000 (1)	1727222040269000.0000	777930486563.2500	0.000000 (1)
	400	1727996400439630.0000	1727991944817710.0000	4455621923.0000	0.000000 (1)	1716392835963730.0000	11603564475898.3000	0.000000 (1)
	600	1727970985655110.0000	1727945403715830.0000	25581939288.2500	0.000000 (1)	1679379508439150.0000	48591477215964.8000	0.000000 (1)
	800	1727890440027340.0000	1727805158813020.0000	85281214323.2500	0.000000 (1)	1597662758376130.0000	130227681651214.0000	0.000000 (1)
	1000	1727715593620590.0000	1727460212199730.0000	255381420857.2500	0.000000 (1)	1463152563598520.0000	264563030022069.0000	0.000000 (1)
	1200	1727363193497010.0000	1726773363259150.0000	589830237867.0000	0.000000 (1)	1259639566256750.0000	467723627240265.0000	0.000000 (1)
DTLZ4	200	728999.4140	728999.4078	0.0062	0.277231 (0)	728991.3901	8.0240	0.000000 (1)
	400	728999.4065	728999.3896	0.1154	0.000000 (1)	728201.9349	434.6057	0.000000 (1)
	600	728999.3788	728999.3464	0.0324	0.000000 (1)	720704.2965	8295.0824	0.000000 (1)
	800	728999.3150	728999.1945	0.1206	0.000000 (1)	696046.7161	32952.5989	0.000000 (1)
	1000	728999.1477	728998.6192	0.5285	0.000000 (1)	644641.3868	84357.7609	0.000000 (1)
	1200	728998.5780	728994.9171	3.6609	0.000000 (1)	559363.2154	169635.3626	0.000000 (1)
DTLZ5	200	1727866.0538	1727865.9384	0.1154	0.013007 (1)	1727431.4481	434.6057	0.000000 (1)
	400	1727865.5061	1727864.6278	0.8784	0.000000 (1)	1721336.0150	6529.4911	0.000000 (1)
	600	1727863.4153	1727859.1967	4.2187	0.000000 (1)	1697620.6971	30242.7183	0.000000 (1)
	800	1727857.2346	1727840.7092	16.5255	0.000000 (1)	1635056.7976	92800.4370	0.000000 (1)
	1000	1727837.7148	1727760.3047	77.4101	0.000000 (1)	1510727.5139	217110.2010	0.000000 (1)
	1200	1727773.5677	1727524.1382	249.4296	0.000000 (1)	1313095.5122	414678.0555	0.000000 (1)
DTLZ6	200	999967922.4861	999899450.4162	68472.0699	0.000000 (1)	999330750.1795	637172.3066	0.000000 (1)
	400	998891441.7157	996820925.0570	2070516.6587	0.000000 (1)	987305237.5147	11586204.2010	0.000000 (1)
	600	990768252.6193	981381295.3432	9386957.2761	0.000000 (1)	948509739.2585	42258513.3608	0.000000 (1)
	800	964064335.0353	937687686.8457	26376648.1896	0.000000 (1)	874883514.5826	89180820.4527	0.000000 (1)
	1000	906131328.8124	855654049.3429	50477279.4695	0.000000 (1)	744576613.9787	161554714.8337	0.000000 (1)
	1200	803763312.2166	712087777.8538	91675534.3628	0.000000 (1)	551828946.6234	251934365.5932	0.000000 (1)
DTLZ7	200	2203.4849	2203.4656	0.0193	0.000000 (1)	2055.0598	148.4252	0.000000 (1)
	400	2193.4627	2192.7324	0.7303	0.000000 (1)	1699.4438	494.0190	0.000000 (1)
	600	2090.3379	2067.9036	22.4343	0.000413 (1)	1338.1314	752.2065	0.000000 (1)
	800	1842.2642	1815.7768	26.4875	0.013007 (1)	1059.6383	782.6260	0.000000 (1)
	1000	1605.8489	1526.3840	79.4649	0.000001 (1)	855.6279	750.2210	0.000000 (1)
	1200	1398.8865	1352.2878	46.5987	0.006223 (1)	718.3771	680.5094	0.000000 (1)

Table 1: Average of the hypervolume indicator. 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 that of the other MOEAs. The P(H) columns shows the results of Wilcoxon's rank sum test. 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.

4. 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*, 14(4):618–635, August 2010.
5. 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)*, pages 1893–1900, Hong Kong, June 2008. IEEE Service Center.
6. P. R. Ehrlich and P. H. Raven. Butterflies and Plants: A Study in Coevolution. *Evolution*, 18(4):586–608, 1964.
7. S. Kukkonen and J. Lampinen. GDE3: The third Evolution Step of Generalized Differential Evolution. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)*, volume 1, pages 443–450, Edinburgh, Scotland, September 2005. IEEE Service Center.
8. X. Y. Mohammad Nabi, Zhenyu Yang. Cooperative co-evolution for large scale optimization through more frequent random grouping. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, volume 1, pages 1– 8 Vol.1, sept. 2010.
9. N. Noman and H. Iba. Enhancing differential evolution performance with local search for high dimensional function optimization. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, GECCO 2005, pages 967–974, New York, NY, USA, 2005. ACM.
10. M. A. Potter and K. A. De Jong. A Cooperative Coevolutionary Approach to Function Optimization. In *Parallel Problem Solving from Nature–PPSN III*, pages 249–257, Jerusalem, Israel, October 9-14 1994. Springer-Verlag. Lecture Notes in Computer Science Vol. 866.
11. M. A. Potter and K. A. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, March 2000.
12. F. van den Bergh and A. P. Engelbrecht. A cooperative approach to particle swarm optimization. *Trans. Evol. Comp*, 8(3):225–239, June 2004.
13. Z. Yang, K. Tang, and X. Yao. Multilevel Cooperative Coevolution for Large Scale Optimization. In *2008 IEEE Congress on Evolutionary Computation*, pages 1663–1670. IEEE Press, 2008.
14. Q. Zhang and H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, December 2007.
15. E. Zitzler, D. Brockhoff, and L. Thiele. The Hypervolume Indicator Revisited: On the Design of Pareto-compliant Indicator Via Weighted Integration. In S. O. et al., editor, *Evolutionary Multi-Criterion Optimization, 4th International Conference, EMO 2007*, pages 862–876, Matshushima, Japan, March 2007. Springer. Lecture Notes in Computer Science Vol. 4403.
16. E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.
17. E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, April 2003.