Comparison Study of Large-scale Optimisation Techniques on the LSMOP Benchmark Functions

Heiner Zille*, Sanaz Mostaghim*
*Institute for Intelligent Cooperating Systems
Faculty of Computer Science, Otto von Guericke University Magdeburg, Germany
Email: {heiner.zille, sanaz.mostaghim}@ovgu.de

***Abstract*—In this paper, we study the performance of three popular large-scale optimisation algorithms on the recently proposed large-scale many-objective optimisation problems (LSMOP). We briefly explain the three methods (MOEA/DVA, LMEA and WOF) and give an overview of their use and performance in the literature. For the Weighted Optimization Framework (WOF), we propose a new transformation function to eliminate the parameter needed in its previous version. In our experiments, we compare the three algorithms on the LSMOP1-9 functions with 2 and 3 objectives and up to 1006 decision variables. The special focus of our study is on the convergence speed and behaviour, since MOEA/DVA and LMEA, in contrast to WOF, need huge computational budgets to obtain variable groups prior to optimisation. Our experiments show that MOEA/DVA and WOF perform significantly better than LMEA on almost all instances and WOF further outperforms MOEA/DVA significantly in most of the 1006-variable problems, in solution quality as well as convergence speed. In most instances the WOF only needs 0.1% to 10% of the total evaluations to outperform the final solution sets obtained by LMEA and MOEA/DVA.**

***Index Terms*—Multi-objective optimization, large-scale optimization, variable Grouping, many-variable optimization, meta-heuristic Framework, WOF, LMEA, MOEA/DVA, LSMOP**

I. Introduction

In multi-objective optimisation, a variety of methods have been proposed to deal with different kinds of optimisation problems, among which, evolutionary algorithms and particle swarm optimisation might be the ones most often used. In recent years, with the advancement of the optimisation algorithms, it became possible to solve more complex tasks, and the area of handling a larger number of objective functions and decision variables has become popular. This area is often referred to as *large-scale optimisation*. When the dimensionality of the search space, i.e. the number of variables, is increased, the performance of classical evolutionary computation approaches often deteriorates. Finding optimal solutions in problems with a large number of variables is an ongoing challenge both in single- and multi-objective optimisation. Concepts like Cooperative Coevolution [1] and other approaches [2]–[5] have been proposed in the literature, which usually involve a special mechanism for the division of the variables into multiple *groups*.

As algorithms become more advanced, there is also a special need to test and compare their performance. For this purpose, a set of special problems, called LSMOP (Large-scale many-objective problems) have been introduced. The LSMOP benchmark function were proposed in 2016 by Cheng et al. [6] and are specifically designed to test the search abilities of algorithms in large-scale optimisation.

Although the field of large-scale optimisation has been studied extensively in single-objective optimisation, the area of multi- and many-objective problems with large numbers of variables has only grown popularity within the last few years. Most recently, three different algorithms (MOEA/DVA, LMEA and WOF) to solve multi-objective large-scale problems have been proposed, all of which differ from the classical single-objective coevolutionary methods. These three have been tested on a variety of test problems each, but no study so far showed a direct comparison of these three on the recently introduced LSMOP benchmark functions. In [4], the WOF was not compared with LMEA, and was also never tested on the LSMOP before. Further, LMEA and MOEA/DVA have only been tested on the LSMOP benchmarks on 5-objective instances with 500 decision variables in [5], but it is unclear if the superior performance of the LMEA results from a good way to deal with many-objectives or a good way to deal with high-dimensional search spaces.

In this work we use the newly proposed LSMOP1-9 benchmarks to compare the performance of the *Weighted Optimization Framework* (WOF) that was proposed in [2] and [4] and two other most recent large-scale optimisation algorithms, the MOEA/DVA (Multi-objective Evolutionary Algorithm based on Decision Variable Analysis) [3] and LMEA (Large-scale Many-objective Evolutionary Algorithm) [5]. As a baseline, the original SMPSO [7] is also used in the experiments, since it is used as one of the components of the WOF.

The main contribution of this paper are as follows:

1) We compare the final solution quality of the three methods on the LSMOP benchmarks for different amounts of decision variables.
2) Since especially LMEA and MOEA/DVA both use computationally expensive ways (in terms of used function evaluations) to achieve the variable groups, a special focus of this work is to compare the convergence speed to analyse how the algorithms make use of their computational budget.
3) Additionally, we propose a new parameter-free transformation function for the Weighted Optimization Framework, which can replace the function used in [4] and therefore eliminate one of the parameters needed in the original version.

The focus of this paper lies on many-variable problems, so the experiments concentrate on different numbers of variables (from 46 to 1006) and use only 2 and 3 objective functions. Since a major drawback of many methods is the increased amount of needed function evaluations due to initial variable interaction analyses, our study will concentrate not only on the final solution sets of the four algorithms, but specifically deal with the analysis of convergence speed.

This article is structured as follows. In Section II, we briefly explain the concept of multi-objective optimisation, followed by a brief review of related studies on multi-objective large-scale approaches. Section III will shortly explain the basic concepts of the three used large-scale algorithms and give an overview of their merits, disadvantages and performance in the literature. Regarding the WOF, we also propose a new parameter-free transformation function for the use in the WOF, which will also be explained in Section III. Section IV gives an introduction into the LSMOP benchmark problem. The main goal of this paper, the experimental comparison of the algorithms, will be done in Section V, with a special focus not only on the final solution sets, but also on the convergence speed. Finally, the paper is concluded in Section VI.

II. Background and Related Work

Problems in nature and science often contain multiple conflicting objectives. These problems are called multi-objective problems (MOPs) and can mathematically be formulated as:

$$\begin{aligned} Z: \quad & min \quad \vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), ..., f_m(\vec{x}))^T \\ & s.t. \quad \vec{x} \in \Omega \subseteq \mathbb{R}^n \end{aligned} \tag{1}$$

This kind of MOP maps the decision Space $\Omega = \{\vec{x} \in \mathbb{R}^n | \, \vec{g}(\vec{x}) \leq 0\}$ of dimension n to the objective space of dimension m. A variety of metaheuristic algorithms have been developed to find a non-dominated solution set for approximating the true Pareto-front. Large-scale optimisation usually deals with optimising MOPs that contain a large number (usually ≥ 200) of variables (also called many-variable problems) or a large number (≥ 4) of objective functions (many-objective problems (MaOP)).

In the following we give a brief overview of basic and recent developments in the area of large-scale optimisation. The three most prominent large-scale approaches used in this paper and their performance and related literature are introduced separately in Section III.

Even though a variety of large-scale optimisers have been developed in recent years, as mentioned before most of them concentrate on single-objective optimisation. An overview of existing large-scale global optimisers for single-objective problems can be found in [8]. One of the most popular concepts is *Cooperative Coevolution* (CC), which was first introduced into the area of optimisation by Potter and De Jong in 1994 [1]. CC aims to optimise several independent populations of subsets of the n decision variables. New solution candidates have to be formed by combining the variable values from different subcomponents. However, genetic operators are only applied within each subcomponent population. The concept of CC has since been used in a variety of large-scale single-objective algorithms [9]–[13].

For multi-objective optimisation, the concept of cooperative coevolution was adapted in the CCGDE3 algorithm [14], combining CC with the GDE3 optimiser. The CCGDE3 algorithm was tested for the ZDT1-3 and ZDT6 [15] problems with up to 5000 decision variables, and performed well especially for the high-dimensional instances. However, the concept of CC has some drawbacks when applied to multi-objective problems. The same ZDT experiments with up to 5000 variables were repeated in [4], and it was also tested on up to 1000-dimensional problems of the WFG [16], DTLZ [17] and CEC2009 competition (UF) [18] benchmark families. The results showed that CCGDE3 was significantly outperformed by the WOF and in most instances even by classical (i.e. non-large-scale) optimisation methods.

An earlier approach by Iorio and Li [19] combined the concept of CC with the NSGA-II algorithm, and focused on the ZDT problems for their analysis. However, only small-dimensional instances of 10 and 30 variables were tested.

In [20], an optimisation problem based on a real world application was introduced within a competition at the IEEE Congress on Evolutionary Computation (CEC) 2015. This problem involved up to 4864 decision variables, and its multi-objective version has been used in [21] and [22]. However, the first approach did not treat the problem in a black-box manner and used derivative information from the objective functions. Both approaches performed superior to the baseline provided by the authors of [20], but were not tested any further on established benchmark problems.

Concepts like CC usually require a scheme for variable grouping, to divide the n decision variables into different subcomponents. A simple random grouping mechanism was for instance used in [9]. Consideration for non-separable problems for single objective functions was done in [10], where the interaction of variables is taken into account by a learning mechanism for finding the optimal division of variables. A mechanism called *Differential Grouping* was developed in [23] to find improved divisions of the variables in single-objective CC algorithms. Other concepts and extensions to this approach for variable grouping have also been proposed [24], [25].

For dealing with variable groups in multiple objectives, special mechanisms have been applied (see Section III). In contrast to the grouping mechanism used in single-objective optimisation, which focus solely on interactions between variables, modern multi-objective methods also decompose based on the contribution to diversity of the population or convergence towards the optimal front.

Grouping methods like the Differential Grouping and also the multi-objective methods used in MOEA/DVA and LMEA (see below) have the disadvantage of using a large computational budget for finding the interactions and compose variable groups using this information. More recently, this issue was addressed by [26]. They introduced a random-based dynamic grouping strategy called *RDG* to reduce the number

of evaluations used to obtain suitable variable groups. The proposed grouping method was implemented into the same Framework as the MOEA/DVA algorithm and was able to outperform the original MOEA/DVA on 2- and 3-objective UF and WFG problems with 800 and 1000 decision variables. The experiments used 8 million and 10 million function evaluations respectively. Although their method still used the original decomposition into convergence- and diversity-related variables proposed in [3], the major share of function evaluations that MOEA/DVA uses in the variable interaction analysis were not necessary any more in the new RDG-based grouping. Thus, the optimisation algorithm was able to spend a larger amount for the actual search.

Another approach to large-scale multi-objective optimisation was shown in 2016 in [27], where the authors tried to include special consideration for large-scale problems into existing genetic operators. Variable groups were directly used within a polynomial mutation operator, without any further change to the surrounding algorithm. In particular, the work assumed that once a suitable separation of the variables is found, they can be directly used by applying mutation operations on all variables in a group simultaneously, while at the same time keeping the direction and amount of mutation the same for all of them. The experiments in [27] used different grouping methods, most of which did not need any additional function evaluations, on WFG problems with 1000 variables. They showed that even with random groups, the performance of the optimisation can be improved significantly by just applying the changed mutation operator without any additional budget spent for the grouping phase.

III. Used Algorithms

In this work we will compare three recently developed large-scale optimisation algorithms with each other. Each of them will be explained here very briefly to give a short overview for the later experimental analysis.

A. MOEA/DVA

The MOEA/DVA (Multi-objective Evolutionary Algorithm based on Decision Variable Analysis) was proposed in 2016 by Ma et al. [3]. As previous methods based on cooperative coevolution, it is based on dividing the variables into multiple groups and then optimising the variables in these groups independently. In comparison to previous single-objective optimisation methods and grouping mechanisms, the decision variable analysis in MOEA/DVA was designed to identify not only interaction between variables, but also the contribution of a variable to convergence, diversity or both. By a control-variable analysis and an interaction analysis, the vector $\vec{x}$ is first divided into two groups, one that contains the distance-variables, which are supposed to control the convergence or closeness to the Pareto-optimal solutions, and another one which contains the position-related and mixed variables, which are expected to have an influence of the diversity of the solution set. The distance-related variables are then further divided by an interaction analysis into smaller subgroups of interacting variables.

The actual optimisation phase of the MOEA/DVA starts after these grouping steps are completed. Each group of variables is optimised independently by a suboptimiser, which is based on differential evolution to create offspring values of the selected groups' variables. In the end of the optimisation, a so-called *uniformity optimisation* is carried out, which optimises the original problem as a whole without using groups, to obtain a better spread of solutions.

An advantage of the MOEA/DVA is that it takes into account the special requirements of multi-objective optimisation, i.e. the different goals of diversity and convergence during the search. It was tested in [3] on several test problems of the WFG, DTLZ and UF families with small numbers of decision variables (24 and 30) with 2 and 3 objective functions and showed good performance compared to other algorithms.

The disadvantage of the MOEA/DVA is that it largely depends on the separation of the variables into groups. The mechanism has been shown to work, but its complexity requires it to use a major share of the computational budget for the initial grouping phase. The complexity varies depending on the problem, but rises roughly quadratically with increasing numbers of function evaluations. In [4], MOEA/DVA needed $>$ 8,000,000 function evaluations for analysing the groups of a 1000-variable problem.

The large-scale tests that were performed in the original MOEA/DVA paper used 200 decision variables and were limited to the 2-objective ZDT4 and UF1-6 and the 3-objective DTLZ1, DTLZ3 and UF10 benchmarks. The maximum number of evaluations used were set between 1,200,000 to 3,000,000 depending on the test problem. MOEA/DVA outperformed other popular algorithms in these tests, although it must be noted that none of them was specifically designed for large-scale problems. Instances with more than 200 variables have not been tested. In [4] the MOEA/DVA was compared with the Weighted Optimization Framework (WOF) on 1000-variable instances of the UF1-10 and the WFG1-5 and WFG7 problems with two and three objectives. The results showed that on most of the UF benchmarks, the MOEA/DVA performed better than the WOF, although it needed a large computational budget to do so (MOEA/DVA used over 8,000,000 evaluations just for the initial grouping phase). On the other hand the WOF outperformed MOEA/DVA on all WFG problems and the UF10 problem and used only a small budget ($<$ 1,000,000 evaluations) to do so.

B. LMEA

Similar to the MOEA/DVA, the LMEA (Large-scale Many-objective Evolutionary Algorithm) was designed specifically to deal with problems with many-objectives and large numbers of variables [5]. It uses a clustering approach to decide if variables are related to convergence, diversity, or both. Similarly to MOEA/DVA, the total number of decision variables are first divided into convergence- and diversity-related variables. After that, an interaction analysis is carried out only on the

convergence variables to further divide them into subgroups. The different convergence-related groups are them optimised separately with a convergence optimisation procedure, while a different procedure is used to optimise the diversity-related variables. The convergence-related groups and the diversity-related groups are optimised in turns until the maximum number of evaluations is used up. A drawback of LMEA is, that similar to MOEA/DVA, it requires a very large amount of function evaluations to obtain the variable groups.

The LMEA has been tested in [5] on large-scale instances of the LSMOP benchmarks. 500 variables and 5 objectives were used with 6,800,000 function evaluations. The best performance for the LSMOP4 and LSMOP7 tests were obtained by the MOEA/DVA while NSGA-III performed best on the LSMOP6. In the LSMOP1-3, 5, 8 and 9, LMEA performed best. LMEA was then further compared with other algorithms on selected problems from the DTLZ, WFG and UF families with up to 1000 decision variables and 10 objectives. It was outperformed by MOEA/DVA on some instances DTLZ1 and 3 and all instances of the UF9 and 10 problems. However, LMEA performed best on all instances of DTLZ5 and 6, WFG3 and some instances of DTLZ1, 2, 3 and 7, showing that it can work well on large-scale problems which including many variables and many objectives at the same time.

C. Weighted Optimization Framework

The Weighted Optimization Framework (WOF) was developed recently in [2] and [4] and originally aimed to overcome certain shortcomings of Cooperative Coevolution. The methods essentially tries to reduce the dimensionality of the problem by altering a large share of the decision variables at the same time and by the same amount. This is done through so-called transformation functions, which assign a *weight*-value each to a group of decision variables. These weight-values, the same amount as the number of variable groups, are then subject to change through a separate optimisation step, which alters the original solutions indirectly through applying the weights in the transformation function.

The vector of n decision variables $(x_1, .., x_n)$ is divided into a number of γ groups, and each group is then assigned a new variable w_j, $j = 1, ..\gamma$. If a solution for the optimisation problem is to be evaluated, the values of the decision variables x_i are combined (through a transformation function) with the corresponding weight-variable w_j that belongs to the respective group of x_i. By doing so, a new vector of variables $\vec{w} = (w_1, .., w_\gamma)$ is introduced and can be optimised independently of the values of $\vec{x}$.

The WOF then uses this mechanism in the following way:

- Optimise the variables of $\vec{x}$ with a population-based metaheuristic.
- From the current population of individuals, pick a number of q solutions x'_k, $k = 1, .., q$.
- For each $k = 1, .., q$: keep the values of x'_k fixed and use a metaheuristic to optimise the variables of $\vec{w}$ for the given x'_k. A transformation function $\Psi(\vec{w}, \vec{x}'_k)$ is used for the function evaluation.
- Apply the found weights to the original population of solutions $\vec{x}$.
- Eliminate duplicates and perform a non-dominated sorting process of the union set of solutions from the parent population and the weighted populations.

Through this process, the framework tries to exploit the search abilities of existing metaheuristics by applying them inside the framework, while by the dimensionality reduction from n to γ variables, the algorithm can search in a much smaller space much more efficiently. Since it is not guaranteed that the optimal solutions are included in the smaller search space (due to the grouping of variables which keeps their relative ratios fixed), the optimisation of the weights $\vec{w}$ is alternated with a normal optimisation of the original problem. For the second half of the overall available function evaluations, a normal optimisation is carried out to spread solutions.

The WOF needs a grouping mechanism, a transformation function to apply a solution $\vec{w}$ to a solution $\vec{x}$, and other parameters that define how the solutions are chosen and how many evaluations are used in which phase of the algorithm. A detailed description can be found in [4]. In contrast to the version of the WOF that was used in [4], in this work we propose a new transformation function (see below) to eliminate the need for the parameter p in the transformation function. Additionally, our implementation of WOF differs from [4] by frequently applying a restart on the optimisation algorithm in the second half of the optimisation, which showed to increase performance slightly. The selection of the used solutions x'_k is done in this work by reference directions instead of Crowding Distance.

The Weighted Optimization Framework was used in the literature to solve various benchmarks with different numbers of decision variables. It has been tested in [2] and [4] on the ZDT1-4, ZDT6, WFG1-9, UF1-10 and DTLZ1-7 functions with 2 and 3 objectives and numbers of variables ranging from 40 to 5000. The results showed that WOF performed best when a particle-swarm optimiser (SMPSO [7]) is used as the optimiser for the subproblems. WOF-SMPSO outperformed other algorithms and especially the CCGDE3 algorithm on all tested problems significantly. As mentioned above, it was also compared to MOEA/DVA on 1000-dimensional problems and showed better performance of the WFG and UF10 problems, while MOEA/DVA seemed to return better final solution sets for the remaining UF instances.

D. The new Transformation Function

In this subsection we introduce a new transformation function for the WOF-algorithm. The best transformation function used in [4] was the so-called *p-Value-Transformation*. The major drawback of this function is that its performance is dependent on a parameter p. Although a sensitivity analysis in [4] showed that a value of $p = 0.2$ yields good results, it can be of advantage to eliminate this parameter, so the algorithm is more adaptive when dealing with unknown problem properties. The *p-Value* method further has the drawback that small values of p only cover parts of the domain $[x_{i,min}, x_{i,max}]$,

TABLE I
PROPERTIES OF THE NINE LSMOP BENCHMARKS AS LISTED IN [6].

	Modality	Separability
LSMOP 1	Unimodal	Fully Separable
LSMOP 2	Mixed	Partially Separable
LSMOP 3	Multi-modal	Mixed
LSMOP 4	Mixed	Mixed
LSMOP 5	Unimodal	Fully Separable
LSMOP 6	Mixed	Partially Separable
LSMOP 7	Multi-modal	Mixed
LSMOP 8	Mixed	Mixed
LSMOP 9	Mixed	Fully Separable

while too large values result in infeasible values and have to be repaired afterwards. To eliminate these factors and make the transformation parameter-free, we propose the following transformation function in this paper:

$$\Psi(w_j, x_i') := x_i = \begin{cases} x_i' + (w_j - 1.0) \cdot (x_{i,max} - x_i') & \text{if } w_j > 1.0 \\ x_{i,min} + w_j \cdot (x_i' - x_{i,min}) & \text{if } w_j \leq 1.0 \end{cases}$$

$$w_j \in [0, 2]$$

where $x_{i,min}$ and $x_{i,max}$ are the respective lower and upper bounds of the variable x_i. The function Ψ computes the values of a new transformed solution $\vec{x}$ from the two inputs $\vec{w}$ and $\vec{x}'$, where $\vec{x}'$ is the fixed solution picked within the WOF algorithm to do the weight optimisation, and $\vec{w}$ is the solution to the transformed subproblem as described above. Each new variable $x_i, i = 1, .., n$ is computed between $x_{i,min}$ and $x_{i,max}$ using the value of x_i' as a center point. In the transformed problem, the variables w_i, $i = 1, .., \gamma$ are optimised in the interval $[0, 2]$, where by transformation all values $[0, 1]$ are mapped to the interval $[x_{i,min}, x_i']$ in the original search space, while all values $(1, 2]$ are transformed to $[x_i', x_{i,max}]$.

IV. BENCHMARK PROBLEMS

The LSMOP problems were recently proposed in 2016 [6] to enable researchers to develop new methods for many-variable and many-objective problems. The LSMOP test suite allows to create problems with any numbers of decision variables and objective functions. In comparison with the DTLZ, WFG or ZDT problems, which were already examined in the original paper of the Weighted Optimisation Framework (WOF), the LSMOP problems address some shortcomings of previous benchmark functions. They allow to specify the separation into groups und the interactions between the groups beforehand. As a result, the nine LSMOP are meant to test specifically many-objective and many-variable optimisation algorithms. The properties of the benchmarks have been summarised in Table I.

V. EVALUATION

So far in the literature, the WOF has been evaluated extensively of problems of the WFG, DTLZ, ZDT and UF benchmarks, and WOF and MOEA/DVA have been compared to each other in [4]. However, WOF has never been tested on the LSMOP before, and its performance has not been compared to LMEA. As mentioned earlier, LMEA and MOEA/DVA have been compared in [5] on the LSMOP problems, but only in many-variable instances with 5 objectives and only with 500 decision variables, where it remains unclear if the superior performance of the LMEA results from a good way to deal with many-objectives or a good way to deal with high-dimensional search spaces. In most studies, a very large computational budget was used, to enable the algorithms to find the variable groups before optimising. This article will now compare the three most prominent large-scale methods WOF, MOEA/DVA and LMEA on the recently proposed LSMOP benchmarks with up to 1000 variables. We will specifically pay attention to the convergence speed and necessary function evaluations, as this is a major drawback of MOEA/DVA and LMEA. Due to limited space, and also to look into the ability to search high-dimensional search spaces efficiently, the experiments in this paper are limited to 2- and 3-objective instances.

TABLE II
SETTINGS OF THE EXPERIMENTS.

$n = 2$ Objectives		$n = 3$ Objectives	
Variables	Max. Evaluations	Variables	Max. Evaluations
46	400,000	52	400,000
106	1,000,000	112	1,000,000
206	2,000,000	212	2,000,000
1006	10,000,000		

A. Experiment Settings

We use SMPSO, WOF-SMPSO, MOEA/DVA and LMEA to solve the LSMOP1 - 9 problems as proposed in [6]. The number of decision variables have been set to 40, 100, 200 and 1000. Due to the nature of the LSMOP, the actual number of decision variables increases slightly from these numbers as listed in Table II. We perform the experiments with 2 and 3 objective functions. As a stopping criterion we use a maximum number of function evaluations. Although the WOF has been shown to deliver good performance after mostly just $100,000$ function evaluations, the LMEA and MOEA/DVA algorithms usually need a lot of resources in the beginning to perform the detection of the groups and interacting variables. The total numbers of function evaluations have therefore been set sufficiently large, and the analysis will include specifically a convergence analysis with respect to the resources. The used maximum evaluations for the different numbers of variables and objectives are shown in Table II.

For implementation, we used the PlatEMO framework [28] version 1.1. The size of the populations is set to 100. WOF uses a standard linear grouping method like also seen in [14] and [4], where the the variables are divided in evenly sized groups by their index number in increasing order. The number of groups is set to $\gamma = 4$ as in [4]. The distribution index used in all operators is set to 20.0 and the probabilities for Crossover and Mutation are set to 1.0 and $1/n$. For all other parameters, the standard values of the framework are used. In particular, the parameters *NCA* and *NIA* in MOEA/DVA are set to 20 and 6, respectively and the parameters *nSel*, *nPer* and *nCor* of LMEA are set to 5, 50 and 5, respectively.

TABLE III

MEDIAN AND IQR VALUES OF THE RELATIVE HYPERVOLUME FOR THE 2- AND 3-OBJECTIVE PROBLEM INSTANCES AT THE END OF THE OPTIMISATION. ALGORITHMS: S = SMPSO, M = MOEA/DVA, L = LMEA, W = WOF-SMPSO. AN ASTERISK INDICATES STATISTICAL SIGNIFICANCE TO THE RESPECTIVE BEST PERFORMANCE MARKED IN BOLD. WORST VALUES ARE SHOWN IN ITALIC FONT.

		m = 2				m = 3		
		n = 46	n = 106	n = 206	n = 1006	n = 52	n = 112	n = 212
LSMOP1	S	0.997158 (0.000369) *	0.995913 (0.000581) *	0.994050 (0.000675) *	*0.612617* (0.013406) *	*0.935828* (0.028215) *	*0.694618* (0.065900) *	*0.537471* (0.199957) *
	M	**0.998416** (0.000155)	**0.998360** (0.000077)	**0.998293** (0.000076)	0.868933 (0.002032) *	**0.988577** (0.000026)	**0.994339** (0.000257)	**0.994246** (0.000373)
	L	*0.991728* (0.002916) *	*0.991257* (0.003259) *	*0.987305* (0.003954) *	0.952396 (0.010203) *	0.981659 (0.022233) *	0.981506 (0.011202) *	0.979635 (0.012336) *
	W	0.997246 (0.000227) *	0.996246 (0.000346) *	0.994899 (0.000373) *	**0.989185** (0.000512)	0.975265 (0.004027) *	0.968194 (0.004572) *	0.965156 (0.008779) *
LSMOP2	S	*0.977772* (0.007519) *	0.940149 (0.010909) *	0.959985 (0.000671) *	0.989577 (0.000169) *	0.938898 (0.011117) *	*0.963574* (0.003575) *	*0.976048* (0.002034) *
	M	**0.995867** (0.002605)	**0.997297** (0.001608)	0.991646 (0.004827) *	0.990796 (0.012254) *	*0.929616* (0.011518) *	0.976347 (0.001372)	0.984112 (0.000824)
	L	0.980717 (0.032344) *	*0.927805* (0.067462) *	*0.943442* (0.042121) *	*0.984772* (0.002236) *	**0.988458** (0.008301)	**0.982496** (0.021853)	**0.984515** (0.006158)
	W	0.995365 (0.000690)	0.994914 (0.000267) *	**0.995519** (0.000291)	**0.995901** (0.000424)	0.966246 (0.005548) *	0.975680 (0.002984)	0.982336 (0.001909)
LSMOP3	S	— (0.019031) *	*0.026166* (0.106319) *	*0.098232* (0.221685) *	0.069957 (0.379810) *	— (0.009068) *	— (—) *	— (—) *
	M	0.571478 (0.029143) *	0.571483 (0.024274) *	0.544579 (0.035757) *	0.308656 (0.013741) *	0.835941 (0.027178) *	**0.830749** (0.038641)	**0.822223** (0.020408)
	L	0.588475 (0.206361) *	0.433682 (0.210642) *	0.484623 (0.041020) *	— (—) *	**0.887771** (0.045559)	0.808182 (0.062720)	0.782965 (0.042430) *
	W	**0.752289** (0.148909)	**0.843565** (0.302801)	**0.852249** (0.050662)	**0.854107** (0.014353)	0.709349 (0.147606) *	0.578484 (0.077048) *	0.563032 (0.068378) *
LSMOP4	S	0.978572 (0.002372) *	0.985923 (0.000548) *	0.989931 (0.000250) *	*0.979610* (0.003200) *	*0.837533* (0.093105) *	*0.789739* (0.039416) *	*0.889981* (0.007085) *
	M	*0.902537* (0.129581) *	0.992031 (0.001571) *	0.994360 (0.143174) *	0.984978 (0.134924) *	**0.975119** (0.011098)	**0.982402** (0.004398)	**0.984321** (0.004839)
	L	0.930420 (0.026290) *	*0.948497* (0.007787) *	*0.968748* (0.004595) *	0.986383 (0.003790) *	0.969127 (0.005543) *	0.974915 (0.005478) *	0.980871 (0.003371) *
	W	**0.996395** (0.000445)	**0.995942** (0.000278)	**0.995881** (0.000337)	**0.991220** (0.000278)	0.953765 (0.008068) *	0.928247 (0.011427) *	0.947201 (0.006167) *
LSMOP5	S	0.997836 (0.000185)	0.998267 (0.001463)	**0.998352** (0.000132)	0.947808 (0.012106) *	0.889906 (0.057415) *	0.889492 (0.028633) *	0.859168 (0.008126) *
	M	0.933371 (0.065122)	**0.998356** (0.000065)	0.998164 (0.000073) *	**0.997031** (0.000366)	**0.970910** (0.000041)	**0.992158** (0.000219)	**0.991968** (0.000164)
	L	*0.623078* (0.342157) *	*0.622537* (0.354949) *	*0.622269* (0.359490) *	*0.621140* (0.000949) *	*0.809378* (0.812623)	*0.055737* (0.832416) *	— (0.647225) *
	W	**0.997845** (0.000214)	0.997360 (0.000251) *	0.996570 (0.000299) *	0.990235 (0.000974) *	0.970729 (0.008997)	0.969194 (0.009013) *	0.955813 (0.006060) *
LSMOP6	S	0.499103 (0.198168) *	0.635998 (0.042742) *	0.737872 (0.067805) *	0.894227 (0.048292) *	— (0.067727) *	— (—) *	— (0.074227) *
	M	*0.089170* (0.644308) *	0.732793 (0.114792) *	*0.453184* (0.238022) *	*0.518881* (0.083398) *	0.567694 (0.398052) *	0.501164 (0.150629) *	0.312725 (0.173587) *
	L	0.567038 (0.368610) *	*0.421774* (0.180308) *	0.494176 (0.278794) *	0.589384 (0.252248) *	0.481575 (0.318471) *	0.287964 (0.186090) *	0.139251 (0.100901) *
	W	**0.973556** (0.015007)	**0.981482** (0.005501)	**0.983564** (0.004066)	**0.985220** (0.001209)	**0.751064** (0.060560)	**0.648079** (0.072806)	**0.568931** (0.012380)
LSMOP7	S	0.502104 (0.185170) *	0.236499 (0.143781) *	0.172306 (0.476852) *	0.329575 (0.116564) *	— (0.061470) *	— (—) *	*0.001773* (0.324819) *
	M	0.348081 (0.123163) *	*0.039310* (0.085930) *	— (—) *	— (—) *	0.441201 (0.108046) *	0.450487 (0.095433) *	0.465989 (0.218911) *
	L	*0.332273* (0.095046) *	0.121544 (0.024365) *	0.068287 (0.021105) *	0.017610 (0.022917) *	— (—) *	— (0.170825) *	0.002914 (0.041140) *
	W	**0.664462** (0.131310)	**0.593654** (0.200199)	**0.520557** (0.177585)	**0.633649** (0.041245)	**0.716272** (0.067339)	**0.574404** (0.055991)	**0.580269** (0.217147)
LSMOP8	S	0.979034 (0.011255) *	0.973785 (0.006079) *	0.975941 (0.002743) *	0.983706 (0.000817) *	0.930338 (0.027131) *	*0.927459* (0.006752) *	*0.947592* (0.002899) *
	M	0.994860 (0.003809) *	0.995512 (0.001678)	*0.916581* (0.004540) *	*0.928015* (0.069875)	**0.978041** (0.002541)	**0.973636** (0.001324)	**0.978943** (0.001480)
	L	*0.925075* (0.003730) *	*0.948258* (0.003115) *	0.968703 (0.002316) *	0.981594 (0.001470) *	*0.911535* (0.084177)	0.939590 (0.015305) *	0.963754 (0.001969) *
	W	**0.996535** (0.001177)	**0.996159** (0.003013)	**0.986162** (0.010283)	**0.990927** (0.001050)	0.970568 (0.003431) *	0.969253 (0.003122) *	0.973220 (0.005922) *
LSMOP9	S	**0.994420** (0.002451)	0.967781 (0.027580) *	0.938192 (0.027004)	0.968416 (0.000739) *	0.874485 (0.106297)	0.647234 (0.092079) *	0.743362 (0.029852) *
	M	0.860342 (0.169826)	**0.998427** (0.000339)	**0.998372** (0.168985)	0.985415 (0.132166) *	**0.896571** (0.092994)	**0.987458** (0.000394)	**0.987877** (0.000228)
	L	0.833473 (0.028259) *	0.837786 (0.021204) *	0.841871 (0.012712)	*0.843816* (0.003744) *	*0.590870* (0.217942) *	*0.568788* (0.030138) *	*0.576258* (0.144947) *
	W	*0.779446* (—) *	*0.779446* (—) *	*0.779446* (—) *	**0.991194** (0.069633)	0.726906 (0.000000) *	0.726906 (0.000000) *	0.726906 (0.000000) *

For each experiment we perform 21 independent runs and report the median and IQR values of the relative hypervolume [29] indicator. The relative hypervolume is the hypervolume obtained by a solution set in relation to the hypervolume obtained by a sample of the Pareto-front of the problem. The used reference point for the indicator is obtained by using the nadir point of our Pareto-front sample and multiply it by 2.0 in each dimension. This is done to make sure most of the obtained solutions can contribute to the HV, even when the sets are not close to the optimal front. Statistical significance is tested using a Mann-Whitney-U Test and significance is assumed for a value of $p < 0.01$.

For comparing the final solution sets, we report the final hypervolume values after the complete amount of evaluations (see Table II) in Table III. For the comparison of convergence speed, we show the obtained hypervolumes throughout the search, after $10,000$, $100,000$, $200,000$ $500,000$, 1 Million, 2 Million, 4 Million, 7 Million and 10 Million function evaluations for the 2-objective problems with $n = 1006$ variables.

B. Results - Final Solutions

First of all we will take a look at Table III, where we see the final results for 2 and 3 objectives and the different numbers of variables. The results shown in these tables are obtained after the total amount of evaluations as given in Table II. For small numbers of variables with $n = 46$ for 2 objectives best performance is obtained by WOF in 6 out of 10 problems. MOEA/DVA perfroms best in 2 problems, while on the LSMOP9, the original SMPSO gives the best results. For the 3-objective results with $n = 52$ however, MOEA/DVA performs best in 5 problems, while WOF and LMEA can outperform the others in 2 problems each.

When the amount of variables is increased, we can see for $n = 206$ and $n = 212$ that the best performance is usually achieved by either the WOF or MOEA/DVA, where it can be observed that in 3 objectives, MOEA/DVA performs stronger than in 2 objectives, outperforming the others in 7 out of 9 problems. Striking are especially the results of LMEA. The implementation of the LMEA in the PlatEMO framework does not stop the algorithm in between the loops for the subcomponents' optimisation, so LMEA used in most cases are larger budget than the other three algorithms. For the 1006-variable instances, LMEA used roughly 10,800,000 evaluations, while the other three methods stopped after 10 Million evaluations. Even given this fact, and given that LMEA performed much better than MOEA/DVA on the 5-objective

TABLE IV

RELATIVE HYPERVOLUME VALUES (RESPECTIVE MEDIAN RUNS) AFTER DIFFERENT AMOUNTS OF FUNCTION EVALUATIONS. K = 1000, M = 1,000,000. THE PROBLEM INSTANCES HAD 2 OBJECTIVES AND 1006 VARIABLES. VALUES EQUAL TO ZERO ARE SHOWN AS DASHES.

$m = 2, n = 1006$		10k	100k	200k	500k	1M	2M	4M	7M	10M
LSMOP1	SMPSO	0.020608	0.027808	0.037316	0.073282	0.103909	0.213880	0.327381	0.476864	0.612539
	MOEA/DVA	—	—	—	—	—	0.315173	0.554071	0.749464	0.868822
	LMEA	—	—	—	—	—	—	—	—	0.922687
	WOF	**0.905159**	**0.957538**	**0.960146**	**0.962214**	**0.965050**	**0.970189**	**0.977507**	**0.987268**	**0.989059**
LSMOP2	SMPSO	0.987095	0.987555	0.987749	0.987930	0.988550	0.988831	0.989187	0.989389	0.989451
	MOEA/DVA	—	0.978912	0.979006	0.979279	0.979887	0.980620	0.981880	0.981952	0.990670
	LMEA	—	—	—	—	—	0.975905	0.983959	0.984492	0.984645
	WOF	**0.994938**	**0.996119**	**0.996113**	**0.995294**	**0.993867**	**0.994832**	**0.996170**	**0.995942**	**0.995774**
LSMOP3	SMPSO	—	—	—	—	—	—	—	—	0.069949
	MOEA/DVA	—	—	—	—	—	—	—	0.118299	0.292623
	LMEA	—	—	—	—	—	—	—	—	—
	WOF	**0.009615**	**0.442065**	**0.465900**	**0.471928**	**0.480294**	**0.643799**	**0.711779**	**0.825147**	**0.853998**
LSMOP4	SMPSO	0.972864	0.973209	0.973127	0.973519	0.973972	0.974535	0.977729	0.979514	0.979485
	MOEA/DVA	—	0.781135	0.785679	0.799552	0.818743	0.836755	0.847009	0.849992	0.984852
	LMEA	—	—	—	—	—	—	—	—	0.983399
	WOF	**0.987622**	**0.990040**	**0.990261**	**0.988011**	**0.988532**	**0.989846**	**0.990470**	**0.991156**	**0.991093**
LSMOP5	SMPSO	—	—	—	—	—	—	—	0.624287	0.947691
	MOEA/DVA	—	—	—	—	—	—	0.396458	0.757481	**0.996907**
	LMEA	—	—	—	—	—	—	—	0.604419	0.620505
	WOF	**0.849223**	**0.897406**	**0.899551**	**0.913478**	**0.924259**	**0.939500**	**0.960709**	**0.984547**	0.990112
LSMOP6	SMPSO	0.600924	0.600751	0.600316	0.600534	0.826841	0.852634	0.866587	0.884579	0.894117
	MOEA/DVA	—	—	—	—	—	—	—	—	0.518592
	LMEA	—	—	—	—	—	—	0.583396	0.586062	0.589142
	WOF	**0.619623**	**0.945416**	**0.948054**	**0.949394**	**0.951907**	**0.955570**	**0.965410**	**0.981979**	**0.985098**
LSMOP7	SMPSO	—	—	—	—	—	—	—	—	0.329535
	MOEA/DVA	—	—	—	—	—	—	—	—	—
	LMEA	—	—	—	—	—	—	—	—	0.017372
	WOF	**0.252052**	**0.344311**	**0.344321**	**0.344321**	**0.344321**	**0.475189**	**0.475297**	**0.505283**	**0.633570**
LSMOP8	SMPSO	—	—	—	—	—	—	0.873835	0.972120	0.983584
	MOEA/DVA	—	—	—	—	—	—	0.386293	0.807462	0.927397
	LMEA	—	—	—	—	—	—	—	0.963716	0.980153
	WOF	**0.830327**	**0.907935**	**0.911661**	**0.920578**	**0.930974**	**0.950856**	**0.969057**	**0.987413**	**0.990804**
LSMOP9	SMPSO	0.099343	0.168975	0.221277	0.438552	**0.964165**	**0.965278**	**0.967271**	0.967816	0.968320
	MOEA/DVA	—	—	—	—	—	0.215045	0.655475	0.807842	0.985326
	LMEA	—	—	—	—	—	—	—	—	0.836770
	WOF	**0.779420**	**0.800507**	**0.814752**	**0.834644**	0.842857	0.847785	0.850620	**0.990224**	**0.991104**

instances in [5], it is outperformed significantly by the other methods in our experiments in almost all problem instances.

To examine especially the performance for large-scale instances, we are now looking on the 6th column of Table III, with $n = 1000$ variables. In the final obtained hypervolume values, WOF performed significantly better than MOEA/DVA and LMEA in 7 out of 9 problems, is on par with MOEA/DVA on the LSMOP8, and performs worse than it only on the LSMOP5 problem.

Overall, when we take a look at the properties of the different LSMOP functions, in Table I, we see that although the WOF is the only of the three large-scale methods which does not spend function evaluation on obtaining suitable groups, it can outperform the other methods not only on the fully separable LSMOP functions, but also achieves best hypervolume values on the LSMOP3, 4 and 8, which are neither fully nor partially separable. These results indicate that sophisticated grouping mechanism might return good variable groups, but the function evaluations spent for it might not boost performance as much as spending the complete budget on optimisation, like WOF does.

C. Results - Convergence

In this section we will take a look at the convergence behaviour of the algorithms. To analyse this, we take a closer look at the 2-objective problems with $n = 1006$ decision variables. In Table IV we list the relative hypervolume values that are obtained by each algorithm after certain numbers of function evaluations. The numbers given are the values obtained by the respective run with the median hypervolume at the end of the 10 Million evaluations. The respective best value for each column (per problem) is marked in bold font. Values equal to zero are shown as dashes.

Table IV shows that the WOF has a major advantage in terms of convergence speed compared to the other three methods. After just 10,000 function evaluation, the WOF method already obtained hypervolume values greater than 0 on all nine problems. With the exception of the LSMOP 3 and 7, WOF obtained a relative hypervolume of over 0.8 after just 100,000 evaluations on all problems and a value of > 0.9 of the LSMOP1, 2, 4-6 and 8 problems after 500,000 evaluations. The other three methods do in most cases not obtain any hypervolume values, or values equal to zero before they spent 2 to 4 million function evaluations, since they either still do the variable analysis or did not converge before.

The numbers in Table IV support the main drawback of the MOEA/DVA and LMEA methods: their huge computational overhead to do the variable grouping before the actual optimisation starts. The major advantage of WOF is that it does not need any sophisticated separation into groups to be

able to achieve a high performance. Since the MOEA/DVA and LMEA spend most of the available evaluations for the analysis of variable interactions, the solutions created during that process are not used to optimise the problem and advance towards better solutions. The LMEA has not obtained any positive hypervolume value prior to the 4 Million evaluations mark for the LSMOP 1, 3-5 and 7-9. In the case of LSMOP1, 3, 4, 7 and 9 the LMEA did not obtain any hypervolume before the 10 million mark. The picture of the MOEA/DVA indicates the same issue. MOEA/DVA was not able to achieve any positive hypervolumes before spending 2 to 4 Million evaluations, and with the exceptions of LSMOP5 and 8 is not able to compensate for this in the remainder of the search, yielding worse performance in the end than WOF on 7 out of 9 problems.

In conclusion, the performance of LMEA cannot compete with MOEA/DVA and WOF on almost all test instances. The WOF does perform best significantly on the LSMOP benchmarks overall, in terms of final performance, but also and more importantly in terms of convergence speed. In most instances, WOF only needs 0.1% to 10% of the total evaluations used my MOEA/DVA to obtain suitable solution sets and higher hypervolume values than LMEA and MOEA/DVA at the end of their optimisation.

VI. Conclusion

In this work, we examined the performance of three recent large-scale optimisation algorithms on the LSMOP benchmark suite. We briefly introduced the methods and proposed a new transformation function for the WOF. The results showed that MOEA/DVA and WOF perform significantly better than LMEA, and that WOF outperforms MOEA/DVA significantly on the 1006-variable 2-objective problems, in solution quality as well as convergence. Open work for the future is to tackle the area of many-objective optimisation. LSMOP can also be used as many-objective benchmarks, which is an important topic for future work. LMEA, although significantly inferior to the other methods in our 2- and 3-objective experiments, already solved 5- and 10-objective problems in their original paper, while WOF and MOEA/DVA still need to be tested on many-objective instances.

References

[1] M. A. Potter and K. A. Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature PPSN III*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1994, vol. 866, pp. 249–257.

[2] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "Weighted optimization framework for large-scale multi-objective optimization," in *Companion of Genetic and Evolutionary Computation Conference - GECCO*, 2016.

[3] 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, 2016.

[4] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multi-objective optimization based on problem transformation," *IEEE Transactions on Evolutionary Computation*, Accepted 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7929324/

[5] X. Zhang, Y. Tian, Y. Jin, and R. Cheng, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2016.

[6] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–14, 2016.

[7] A. J. Nebro, J. J. Durillo, J. Garcia-Nieto, C. A. Coello Coello, F. Luna, and E. Alba, "SMPSO: A new PSO-based metaheuristic for multi-objective optimization," in *Symposium on Computational Intelligence in Multi-criteria Decision-making*. IEEE, 2009, pp. 66–73.

[8] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Metaheuristics in large-scale global continues optimization: A survey," *Information Sciences*, vol. 295, pp. 407–428, 2015.

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

[10] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving from Nature, PPSN XI*, ser. Lecture Notes in Computer Science, R. Schaefer, C. Cotta, J. Kołodziej, and G. Rudolph, Eds. Springer Berlin Heidelberg, 2010, vol. 6239, pp. 300–309.

[11] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2009, pp. 1546–1553.

[12] ——, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 210–224, 2012.

[13] Z. Yang, J. Zhang, K. Tang, X. Yao, and A. C. Sanderson, "An adaptive coevolutionary differential evolution algorithm for large-scale optimization," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2009, pp. 102–109.

[14] L. M. Antonio and C. A. Coello Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 2758–2765.

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

[16] 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, 2006.

[17] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *IEEE Congress on Evolutionary Computation (CEC)*, vol. 1, 2002, pp. 825–830.

[18] Q. Zhang, A. Zhou, and S. Zhao, "Multiobjective optimization test instances for the CEC 2009 special session and competition," *University of Essex*, 2008.

[19] A. W. Iorio and X. Li, "A cooperative coevolutionary multiobjective algorithm using non-dominated sorting," in *Genetic and Evolutionary Computation Conference - GECCO*. Springer, 2004, pp. 537–548.

[20] S. K. Goh, K. C. Tan, A. Al-Mamun, and H. A. Abbass, "Evolutionary big optimization (BigOpt) of signals," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 3332–3339.

[21] Y. Zhang, J. Liu, M. Zhou, and Z. Jiang, "A multi-objective memetic algorithm based on decomposition for big optimization problems," *Memetic Computing*, vol. 8, no. 1, pp. 45–61, 2016.

[22] S. Elsayed and R. Sarker, "An adaptive configuration of differential evolution algorithms for big data," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 695–702.

[23] M. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 378–393, 2014.

[24] M. N. Omidvar, Y. Mei, and X. Li, "Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2014, pp. 1305–1312.

[25] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in *Genetic and Evolutionary Computation Conference - GECCO*. ACM Press, 2015, pp. 313–320.

[26] A. Song, Q. Yang, W. N. Chen, and J. Zhang, "A random-based dynamic grouping strategy for large scale multi-objective optimization," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, July 2016, pp. 468–475.

[27] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "Mutation operators based on variable grouping for multi-objective large-scale optimization," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, Dec 2016.

[28] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "Platemo: A MATLAB platform for evolutionary multi-objective optimization," *CoRR*, vol. abs/1701.00879, 2017. [Online]. Available: http://arxiv.org/abs/1701.00879

[29] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms a comparative case study," *Parallel Problem Solving from Nature PPSN V*, 1998.