2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Cooperative Co-evolution with a New Decomposition Method for Large-Scale Optimization

Sedigheh Mahdavi
Department of Mathematics and Computer Science
Amirkabir University of Technology
Tehran, Iran
s_mahdavi@aut.ac.ir

Mohammad Ebrahim Shiri
Department of Mathematics and Computer Science
Amirkabir University of Technology
Tehran, Iran
shiri@aut.ac.ir

Shahryar Rahnamayan
Faculty of Engineering and Applied Science of Ontario
Institute of Technology (UOIT)
2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada
Shahryar.Rahnamayan@uoit.ca

Abstract*—Cooperative Co-evolutionary algorithms are effective approaches to solve large-scale optimization problems. The crucial challenge in these methods is the design of a decomposition method which is able to detect interactions among variables. In this paper, we proposed a decomposition method based on High Dimensional Model Representation (HDMR) which extracts separable and nonseparable subcomponents for Cooperative Co-evolutionary algorithms. The entire decomposition procedure is conducted before applying the optimization. The experimental results for D=1000 on twenty CEC-2010 benchmark functions show that the proposed method is promisingly efficient to solve large-scale optimization problems. The proposed approach is compared with two other methods and discussed in details.

I. INTRODUCTION

THE real-world optimization problems are faced with several challenging features, such as, landscape's shape complexity, high dimensionality, nonlinearity, concavity, uncertainty, and expensive objective evaluation. Many metaheuristic algorithms have been proposed to tackle these kinds of problems. The performance of these algorithms severely deteriorates when the dimension of the search space increases. Cooperative Coevolution (CC) algorithms [1, 2] have been proposed to handle the problems with a large number of decision variables. These algorithms consider decomposition methods to tackle a high dimensional problem. In the classical CC method, a high-dimensional decision vector is divided into some low-dimensional subcomponents/vectors. The classical CC methods are inefficient for handling nonseparable problems since they do not take into account to identify the interactions among variables. In separable problems, there is no interaction among variables; therefore it is possible to decompose it easily into several low-dimensional subcomponents. But, the nonseparable problems consist of interacting subcomponent variables. The efficiency of the CC algorithms deteriorate when interacting variables are placed in different subcomponents, because the influence of a variable on the fitness value in one subcomponent depends on other variables in different subcomponents; and subcomponents cannot be evolved independently. The ideal grouping method in CC algorithms should decompose a high dimensional problem into several subcomponents such that the interactions among different subcomponents are minimal [3, 4]. It was shown that decomposition methods have a significant influence in the performance of the CC algorithms [3]. Therefore, the crucial challenge in the success of CC algorithms is developing a suitable decomposition method with a higher interaction recognition success rate.

In recent years, several decomposition methods for the CC algorithms have proposed to identify interacting variables for constructing the subcomponents of a large-scale problem. Two major categories of these methods, in term of the variable grouping strategy, are static and dynamic decomposition methods. In the static decomposition methods, the associated variables to each subcomponent are kept fixed during the optimization process while in the dynamic decomposition methods, grouping changes during the optimization.

The High Dimensional Model Representation (HDMR) [14, 15] is a general set of quantitative model assessment and analysis tool which has been introduced to capture high-dimensional input– output system behavior. HDMR represents a map of the relationship between input and output system variables. Since the general assumption in the CC methods is that there is no available information about the problem (i.e., black-box function), it would be useful to obtain some significant knowledge of a problem via HDMR for discovering unknown better arrangement of subcomponents.

In the paper, a new decomposition method for the CC algorithms is proposed based on HDMR method (DM-HDMR) which discovers nonseparable and separable subcomponents to improve the performance of CC algorithms. First, we approximate a black-box function using a RBF-HDMR approach, proposed by Shan and Wang [5], and then the interactions among variables based on the first order RBF-HDMR are detected and the separable and nonseparable subcomponents are constructed. The DM-HDMR is tested on

20 high-dimensional (i.e., D=1000) benchmark functions. The obtained results indicate that the proposed approach is highly efficient.

The organization of the rest of this paper is as follows. Section II presents a background review. Section III describes the details of the proposed decomposition method. Section IV presents the experimental results and discussion. Finally, the paper is concluded in Section V.

II. BACKGROUND REVIEW

A. Decomposition Methods

Potter and De Jong [2] proposed two basic CC algorithms in 1994, they proposed the one-dimensional based strategy and splitting-in-half decomposition strategy. In one-dimensional based strategy, an n-dimensional problem is decomposed into n one-dimensional subcomponents. In each subcomponent, the fitness of an individual is computed by an n-dimensional vector which is constructed by this individual and the selected members of other subcomponents. In the second strategy, an n-dimensional problem is decomposed into two n/2 subcomponents. It was confirmed that the performance of CC algorithms depend on identifying interactions among variables and the efficiency of the methods drops when solving nonseparable problems [6].

Recently, a random decomposition method (DECC-G) was introduced by Yang et al. to handle the high-dimensional nonseparable problems (tested up to 1000D) [4]. In the random decomposition method, an n-dimensional problem is decomposed randomly into multiple low-dimensional subcomponents. They used an adaptive weighting strategy to obtain better solutions. When the number of interacting variables grows, the performance of DECC-G method decreases. In [7], a self-adaptive decomposition method (MLCC) was proposed. MLCC utilizes different subcomponent sizes of a decomposer pool to identify variant interaction levels. A drawback of MLCC is that the problem is decomposed into a set of equal size subcomponents.

The correlation matrix based methods were proposed to form subcomponents [8, 9]. These algorithms calculate correlations among the variables in each generation and variables are placed in the separate subcomponents according to the computed correlations. In these methods, the huge computational resources are used and the nonlinear dependencies among the variables are not detected. In [10], the delta decomposition method was proposed in which a problem is decomposed based on the absolute amount of change in each dimension at two sequential cycles. When the problem includes multiple nonseparable components, the efficiency of this method deteriorates accordingly. Chen et al. [11] introduced a CC method with Variable Interaction Learning (CCVIL) to determine the structure of subcomponents and to adjust the component size. They incorporated a learning stage in CC algorithm which is capable to detect the interactions among the decision variables. In [12, 13] , the dependency identification (DI) techniques were proposed; they define an internal minimization problem according to the concept of partially separable function during the CC algorithm. Omidvar et al. [3] introduced an automatic decomposition approach (DECC-DG), differential grouping, based on the description of the partial separable functions. In DECC-DG method, the nonseparable subcomponents are identified in a pairwise fashion. A theorem is defined and proved to identify the interactions among two variables. This theorem shows that if the change in the objective function with respect to one variable depends on another variable, then two variables are nonseparable.

B. High Dimensional Model Representation (HDMR)

A general form of HDMR is given as follows [14, 15]:

$$f(x_1,\ldots,x_N)=f_0+\sum_{i_1=1}^{N} f_{i_1}(x_{i_1}) + \sum_{\substack{i_1,i_2=1\\ i_1<i_2}}^{N} f_{i_1 i_2}(x_{i_1},x_{i_2})+\cdots+f_{123..N}(x_1,\ldots,x_N)$$

where the component f_0 is a constant representing the zero-th order effect to $f(x)$; the function $f_i(x_i)$ is a first order term giving the effect of variable x_i acting independently upon the output $f(x)$; the function $f_{ij}(x_i, x_j)$ is a second order term describing the cooperative effects of the variables x_i and x_j upon the output $f(x)$. The higher order terms reflect the cooperative effects of increasing numbers of input variables acting together to influence the output $f(x)$.

We apply the RBF-HDMR which was recently introduced by Shan and Wang [5]. They used a type of RBF function which uses a sum of a thin plate spline function (the first term) and a linear polynomial $P(x)$ (the second term) to approximate each component function in the special class of HDMR, i.e., Cut-HDMR. A general RBF-HDMR is given by:

$$f(x_1,\ldots,x_N)=f_0+\sum_{i=1}^{N}\sum_{k=1}^{m_i}\alpha_{i_k}\left|(x_i,x_0^i)-(x_{i_k},x_0^i)\right|+\sum_{1\le i<j\le d}\sum_{k=1}^{m_{ij}}\alpha_{ij_k}\left|(x_i,x_j,x_0^{ij})-(x_{i_k},x_{j_k},x_0^{ij})\right|$$
$$+\ldots\ldots.+\sum_{k=1}^{m_{ij}}\alpha_{1,2,\ldots.d_k}\left|x-x_k\right|$$

Where $\alpha_{i_k},\alpha_{ij_k},\ldots,\alpha_{1,2,\ldots\ldots.d_k}$ are the coefficients of the expression and $(x_{i_k},x_0^i),(x_{i_k},x_{j_k},x_0^{ij}),\ldots x_k$ are the sampled points of input variables or the centers of a linear RBF approximation. They proposed strategies to save the cost of constructing higher order components in HDMR. In the second order RBF-HDMR model, there is a cooperative effect of the variables x_i and x_j if the first RBF-HDMR model passes through randomly selected points from the set of sampled points which used to construct the first-order component terms f_i and f_j. Base on the mentioned strategy, we proposed a new decomposition method by using the first RBF-HDMR model.

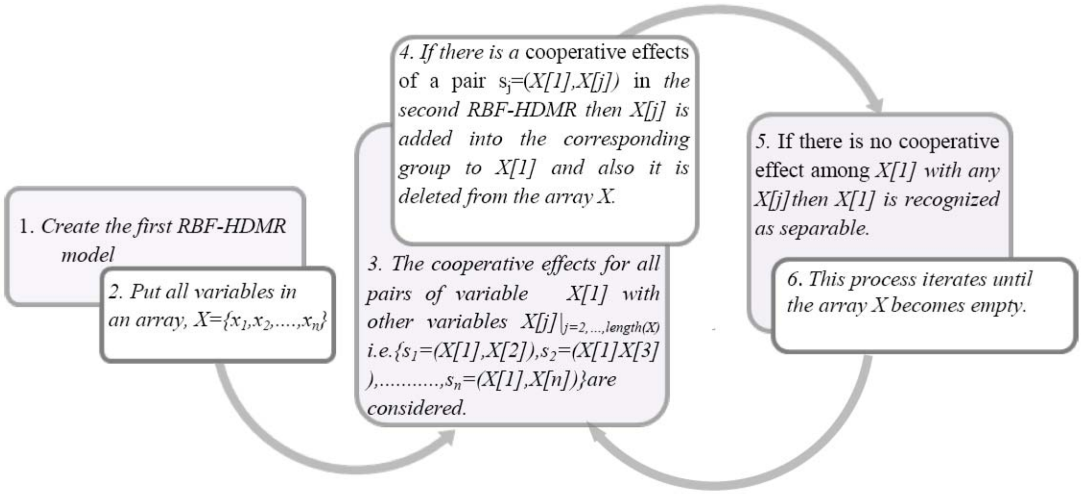


Fig. 1. The simplified flow of the DM-HDMR process

III. PROPOSED DM-HDMR DECOMPOSITION METHOD

A major challenge in CC methods is the design of a suitable decomposition method which can recognize interactions among variables [3, 12]. There are the drawbacks of existing research works in constructing separable and nonseparable subcomponents hence the essential requirement for the design of a decomposition strategy is desirable. In this paper, a decomposition method based on HDMR (DM-HDMR) is proposed which is described as follows. Figure 1 shows a simplified flow of the DM-HDMR process.

The key feature of RBF-HDMR model is modeling high-dimensional problems. We propose the idea of employing RBF-HDMR model to extract interactions among variables. Algorithm1 shows that how the RBF-HDMR model can be used to identify the interacting variables and to create the nonseparable and separable subcomponents. At the beginning of the DM-HDMR algorithm, the first-order RBF-HDMR model is computed based on the mentioned algorithm in [5]. The decomposition step starts by discovering interacting variables with the first variable and then its corresponding subcomponent is formed. In the DM-HDMR algorithm, two variables are recognized as nonseparable if for constructing the second-order RBF-HDMR model a cooperative effect of two variables is identified according to the Shan and Wang's strategy. For variables x_i and x_j, two points are randomly selected from a point set which is formed based on all sample values of the variables x_i and x_j for modeling the first-order term of components f_i and f_j.

In the DM-HDMR algorithm, when an interaction among a variable with first variable is detected, this variable is placed in the first variable subcomponent and then it is removed from the set of all variables. This process is repeated for other remaining variables until there is no more variable left. A variable is recognized as a separable variable if there is no interaction among it with all remaining variables.

Algorithm1*: allgroups* $\longleftarrow$ *decomposition (func, n)*

1. *dims* $\longleftarrow$ *{1,2,....,n}*
2. *sep* $\longleftarrow$ *{}   // contains all separable variables*
3. *Allgroups* $\longleftarrow$ *{}*
   //contains all non-separable subcomponents
4. *Create the first RBF-HDMR(i.e.,*
   $$f(x_1,\ldots,x_N)=f_0+\sum_{i=1}^{d}\sum_{k=1}^{m_i}\alpha_{i_k}\left|(x_i,x_0^i)-(x_{i_k},x_0^i)\right|$$
   according to Shang and Wang' algorithm
5. ***for*** $i \in$ *dims* ***do***
6. *group* $\longleftarrow$ *{}*
7. ***for*** $j \in$ *dims* $\wedge$ $i \neq j$ ***do***
8. *select a random point from the sampled set of points for constructing the first-order component term components* f_i *and* f_j
9. ***if*** *(the first RBF-HDMR model does not pass through a selected random point)* ***then***
10. *group* $\longleftarrow$ *group* $\cup$ *j*
11. ***end if***
12. ***end for***
13. *dims* $\longleftarrow$ *dims- group*
14. ***if*** *length (group)==1* ***then***
15. *seps* $\longleftarrow$ *seps* $\cup$ *group*
16. ***Else***
17. *allgroups* $\longleftarrow$ *allgroups* $\cup$ *{ group }*
18. ***end if***
19. ***end for***

A. Combining DM-HDMR Decomposition Algorithm with CC

Algorithm 2 explains how the DM-HDMR decomposition algorithm is used in a CC framework. The basic CC framework [1, 2] is applied in this algorithm. In the first step, DM-HDMR constructs nonseparable and separable subcomponents. Then, all subcomponents are optimized in a round-robin fashion by Self-adaptive Differential evolution with neighbourhood search strategy (SaNSDE) [4].

IV. EXPERIMENTAL RESULTS

The twenty benchmark functions are used to evaluate the performance of DM-HDMR method. These functions were provided by the CEC-2010 Special Session and Competition on LSGO [16]. In this benchmark test set, there are five types of functions:

1- Separable functions (f_1-f_3)

2- Single-group m-nonseparable functions (f_4-f_8)

f_4: Single-group Shifted and m -rotated Elliptic Function.

f_5: Single-group Shifted and m -rotated -rotated Rastrigin's Function.

f_6: Single-group Shifted and m -rotated Ackley's Function.

f_7: Single-group Shifted and m-rotated Schwefel's Problem 1.2

f_8: Single-group Shifted and m -rotated Rosenbrock's Function.

3- $\frac{n}{2m}$ -group m -nonseparable functions (f_9-f_{13})

4- $\frac{n}{m}$ - group m -nonseparable functions(f_{14}-f_{18})

5- Nonseparable functions (f_{19}-f_{20})

Where n is the dimensionality of the function and m is the number of variables in each nonseparable subcomponent.

A. Simulation Results

Table I summarizes the experimental results for CCVIL, DECC-DG, and DM-HDMR methods. The grouping accuracy is computed similar to the research work [3] . It can be found that the decomposition accuracy for 12 out of 20 benchmark functions is 100%. In three fully separable functions (f_1-f_3), all variables are placed in a subcomponent. The DM-HDMR algorithm correctly recognized all the decision variables fully separable.

Algorithm2*: allgroups* $\longleftarrow$ *decomposition (func, n)*

```
pop ⟵ initialize (popsize, n)
groups ⟵ DG-HDMR(func, n)
(best,best val) ⟵ evaluate(pop)
for i ⟵ 1 to max_cycle do
  for j ⟵ 1 to length (groups) do
    index_subpop ⟵ groups[j]
    subpop ⟵ pop(: , index_subpop )
    (subpop,new) ⟵ optimizer(best, subpop)
    pop(: , index_subpop ) ⟵ subpop
    best ⟵ new
  end for
end for
```

In single-group m *-nonseparable* functions, the decomposition accuracy for 2 (i.e., f_5, f_6) out of these 5 functions is 100%. On f_7, the algorithm has obtained the nonseparable subcomponent approximately correct with little difference with the real nonseparable subcomponent while it only could not identify six nonseparable variables. For f_4, identifying interactions among variables is poor and the algorithm cannot discover correctly nonseparable subcomponents. The nonseparable variables are found correctly in f_8 but separable variables were misplaced.

For the third type of benchmark functions (f_9-f_{13}), where there are 10 independent 50-nonseparable subcomponents and one separable subcomponent with 500 variables, the decomposition accuracy for 3 functions (f_9, f_{10}, and f_{12}) out of 5 functions is 100%. The performance of DM-HDMR algorithm is significant on f_{11} and most nonseparable variables are discovered.For f_{13}, the captured interacting variables are not correct. In fully nonseparable functions (f_{14}-f_{20}), it has the high performance for all functions except on f_{18} so that the accuracy of interacting variables is 100% for functions f_{14}, f_{17}, and f_{19}-f_{20}.

We have also compared the performance of the DM-HDMR algorithm with two other state-of-the-art methods, namely, CCVIL and DECC-DG methods. The results confirm that the CCVIL achieves better results than DM-HDMR and DECC-DG on f_4 and f_7. For f_1, f_2, and f_5; CCVIL has the same results with DM-HDMR and DECC-DG while the performance of DM-HDMR and DECC-DG is better than CCVIL on other functions. The performance of DM-HDMR and DECC-DG deteriorates significantly on most Rosenbrock functions (f_8, f_{13}, f_{18}) while DM-HDMR obtains the remarkable result on f_{20}. Also, DM-HDMR and DECC-DG have the deficient result on f_4 although the accuracy of DECC-DG is 100% for constructing nonseparable subcomponent. On f_{11} *and* f_{16}, DECC-DG outperforms DM-HDMR. Both DM-HDMR and DECC-DG algorithms have approximately same results on other functions.

TABLE I
NUMERICAL RESULTS OF DM-HDMR, DECC-DG, AND CCVIL
THE SYMBOL '‡' IS USED TO INDICATE WHICH ALGORITHMS HAVE THE SAME RESULT
AND THE BEST VALUE IS FORMATTED IN BOLD.

Fun	Sep. Var.s	Non-sep. Var.s	Decomposition strategy	# Captured Sep. Vars.	# Captured Non-Sep.Var.s	# FE
f_1	1000	0	CCVIL	1000‡	0‡	**69990**
			DECC-DG	1000‡	0‡	1001000
			DM-HDMR	1000‡	0‡	516304
f_2	1000	0	CCVIL	1000‡	0‡	**69990**
			DECC-DG	1000‡	0‡	1001000
			DM-HDMR	1000‡	0‡	567231
f_3	1000	0	CCVIL	938	0‡	1798666
			DECC-DG	1000‡	0‡	1001000
			DM-HDMR	1000‡	0‡	**709487**
f_4	950	50	CCVIL	**957**	43	1797614
			DECC-DG	33	**50**	**14564**
			DM-HDMR	0	2	17642
f_5	950	50	CCVIL	950‡	50‡	1795705
			DECC-DG	950‡	50‡	905450
			DM-HDMR	950‡	50‡	**516300**
f_6	950	50	CCVIL	910	47	1796370
			DECC-DG	950‡	50‡	906332
			DM-HDMR	950‡	50‡	**653926**
f_7	950	50	CCVIL	**951**	**49**	1796475
			DECC-DG	247	34	**67250**
			DM-HDMR	948	47	468479
f_8	950	50	CCVIL	**1000**	0	69842
			DECC-DG	135	46	**23608**
			DM-HDMR	0	**50**	29099
f_9	500	500	CCVIL	583	337	1792212
			DECC-DG	500‡	500‡	270802
			DM-HDMR	500‡	500‡	**150445**
f_{10}	500	500	CCVIL	508	492	1774642
			DECC-DG	500‡	500‡	272958
			DM-HDMR	500‡	500‡	**184431**
f_{11}	500	500	CCVIL	476	491	1774565
			DECC-DG	**501**	**499**	**270640**
			DM-HDMR	503	428	293933
f_{12}	500	500	CCVIL	516	435	1777344
			DECC-DG	500‡	500‡	271390
			DM-HDMR	500‡	500‡	**149824**
f_{13}	500	500	CCVIL	1000	0	69990
			DECC-DG	131	**126**	**49470**
			DM-HDMR	**548**	28	333903
f_{14}	0	1000	CCVIL	150	719	1785975
			DECC-DG	0‡	1000‡	**21000**
			DM-HDMR	0‡	1000‡	24725
f_{15}	0	1000	CCVIL	1000	0	1751241
			DECC-DG	**0**	**1000**	**21000**
			DM-HDMR	1	999	40163
f_{16}	0	1000	CCVIL	1000	0	1751647
			DECC-DG	**4**	**996**	**21128**
			HDMR	6	950	98899
f_{17}	0	1000	CCVIL	1000	0	1752340
			DECC-DG	0‡	1000‡	**21000**
			DM-HDMR	0‡	1000‡	23198
f_{18}	0	1000	CCVIL	1000	0	69990
			DECC-DG	85	**173**	34230
			DM-HDMR	**1**	50	**22952**
f_{19}	0	1000	CCVIL	1000	0	48212
			DECC-DG	0‡	1000‡	**2000**
			DM-HDMR	0‡	1000‡	8960
f_{20}	0	1000	CCVIL	1000	0	1798708
			DECC-DG	42	82	**22206**
			DM-HDMR	**0**	**1000**	22823

Table II presents the detail of the subcomponents on a number of functions which are found by DM-HDMR. The Groups column presents a constructed nonseparable subcomponent by DM-HDMR. The Permutation Groups column shows permutation subcomponents (P1-P20) which includes indices of 50 randomly chosen dimensions and the next column beside it indicates the number of variables belong to each permutation subcomponent.

Table III shows the experimental results for a cooperative co-evolution algorithm with DM-HDMR. The mean values and the standard deviation of best objective value of the 25 runs in each run at 3×10^6 FEs per function. The experimental results were compared with the results of DECC-DG. The symbol '‡' is used to indicate which both algorithms have same results and the best value is highlighted in bold.

The results confirm that in fully separable functions class, the CC method with DM-HDMR gained much better results than DECC-DG on f_1 and f_2. The reason is that DM-HDMR can save a significant number of fitness evaluations in the decomposition step. On f_3, both algorithms have the same results.

In the second class, the CC method with DM-HDMR outperforms DECC-DG on f_4-f_6, while the results of DECC-DG are better than the CC method with DM-HDMR on f_7-f_8. In the third class (f_9-f_{13}), the performance of DM-HDMR is better than DECC-DG on f_9-f_{10},f_{12}-f_{13} although DECC-DG outperforms DM-HDMR on f_{11}. In the fully nonseparable functions, DECC-DG outperforms the CC method with DM-HDMR on f_{15}-f_{17} while the results of the CC method with DM-HDMR are better than DECC-DG on f_{18} *and* f_{20}. The performance of both algorithms is the same on f_{14} *and* f_{19}. It can be seen from Table III that DM-HDMR on f_1-f_2, f_5-f_6, f_9-f_{10}, *and* f_{12} outperforms DECC-DG because by saving the number of fitness evaluations in the decomposition step, DM-HDMR can use more significant number of fitness evaluations in the optimization stage. In the figure 2, the convergence plots for selected function indicated the behavior of both DM-HDMR and DECC-DG algorithms.

TABLE II
The nonseparable subcomponents are constructed by DM-HDMR.

Func	Groups	Group Size	Permutation Groups	#var	Func	Groups	Group Size	Permutation Groups	#var
f_{17}	G01	50	P9	50	f_4	G01	998	P1	48
	G02	50	P7	50				P2	50
	G03	50	P2	50				P3	50
	G04	50	P12	50				P4	50
	G05	50	P4	50				P5	50
	G06	50	P20	50				P6	50
	G07	50	P5	50				P7	50
	G08	50	P18	50				P8	50
	G09	50	P10	50				P9	50
	G10	50	P15	50				P10	50
	G11	50	P1	50				P11	50
	G12	50	P8	50				P12	50
	G13	50	P17	50				P13	50
	G14	50	P11	50				P14	50
	G15	50	P14	50				P15	50
	G16	50	P3	50				P16	50
	G17	50	P13	50				P17	50
	G18	50	P19	50				P18	50
	G19	50	P6	50				P19	50
	G20	50	P16	50				P20	50
f_{11}	G01	43	P4	43		G02	2	P1	2
	G02	47	P6	47	f_{10}	G01	50	P4	50
	G03	48	P3	48		G02	50	P6	50
	G04	46	P1	46		G03	50	P3	50
	G05	45	P5	45		G04	50	P1	50
	G06	37	P9	37		G05	50	P5	50
	G07	45	P2	45		G06	50	P9	50
	G08	27	P8	27		G07	50	P2	50
	G09	47	P7	47		G08	50	P8	50
	G10	39	P10	39		G09	50	P7	50
	G11	2	P8	2		G10	50	P10	50
	G12	2	P6	2	f_5	G01	50	P1	50
	G13	2	P9	2	f_6	G01	50	P1	50

TABLE III
COMPARISON OF CC METHOD WITH EMBEDDED DM-HDMR AND DECC-DG METHODS ON THE CEC-2010 BENCHMARK FUNCTIONS.

Function		DM-HDMR	DECC-DG
f_1	Mean	**2.34e+01**	5.47e+03
	Std	5.23e+01	2.02e+04
f_2	Mean	**4.36e+03**	4.39e+03
	Std	1.97e+02	1.97e+02
f_3	Mean	**1.67e+01**‡	**1.67e+01**‡
	Std	3.03e-01	3.34e-01
f_4	Mean	**6.96e+11**	4.79e+12
	Std	2.12e+11	1.44e+12
f_5	Mean	**1.45e+08**	1.55e+08
	Std	1.78e+07	2.17e+07
f_6	Mean	**1.63e+01**	1.64e+01
	Std	4.08e-01	2.71e-01
f_7	Mean	2.91e+05	**1.16e+04**
	Std	2.26e+05	7.41e+03
f_8	Mean	4.41e+07	**3.04e+07**
	Std	2.44e+07	2.11e+07
f_9	Mean	**5.20e+07**	5.96e+07
	Std	8.21e+06	8.18e+06
f_{10}	Mean	**4.49e+03**	4.52e+03
	Std	1.40e+02	1.41e+02
f_{11}	Mean	1.10e+01	**1.03e+01**
	Std	9.94e-01	1.01e+00
f_{12}	Mean	**1.97e+03**	2.52e+03
	Std	3.15e+02	4.86e+02
f_{13}	Mean	**3.35e+06**	4.54e+06
	Std	6.83e+05	2.13e+06
f_{14}	Mean	**3.41e+08**‡	**3.41e+08**‡
	Std	2.60e+07	2.41e+07
f_{15}	Mean	5.95e+03	**5.88e+03**
	Std	9.11e+01	1.03e+02
f_{16}	Mean	1.24e-06	**7.39e-13**
	Std	1.92e-06	5.70e-14
f_{17}	Mean	4.03e+04	**4.01e+04**
	Std	3.19e+03	2.85e+03
f_{18}	Mean	**8.40e+03**	1.11e+10
	Std	4.35e+03	2.04e+09
f_{19}	Mean	**1.71e+06**‡	**1.74e+06**‡
	Std	1.38e+04	9.54e+04
f_{20}	Mean	**2.45e+06**	4.87e+07
	Std	1.23e+07	2.27e+07

V. CONCLUSION REMARKS

In this paper, we proposed a decomposition method inspired from the High Dimensional Model Representation to discover the optimum grouping of the variables for dividing a high dimensional problem into low subcomponents. Significant information can be obtained by constructing the first order RBF-HDMR which is used to determine a cooperative effect of two variables then the separable and nonseparable subcomponents are recognized according to the detected cooperative effect among variables. The DM-HDMR algorithm suggests a promising idea for decomposition methods to handle large-scale problems within cooperative coevolution methods however the further study and more experiments are required to improve this decomposition method. The performance of the DM-HDMR algorithm is evaluated on the CEC-2010 challenging well-known benchmark functions. The DM-HDMR algorithm obtained comparable good results in the majority of functions. Also, a cooperative co-evolutionary framework with using DM-HDMR method was proposed for tackling large-scale optimization problems. Based on the achieved results, it can be concluded that the algorithm can efficiently solve large-scale optimization problems. In future, we intend to develop strategies based on HDMR to better divide the computational budget in a CC algorithm among subcomponents with respect the main effect of theirs variables on the fitness function. In addition, we are interested in applying DM-HDMR to more large-scale benchmark functions.

REFERENCES

[1] M. A. Potter, "The design and analysis of a computational model of cooperative coevolution," Citeseer, 1997.

[2] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature—PPSN III*, ed: Springer, 1994, pp. 249-257.

[3] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative Co-evolution with Differential Grouping for Large Scale Optimization," *methods,* vol. 3, p. 5.

[4] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences,* vol. 178, pp. 2985-2999, 2008.

[5] S. Shan and G. G. Wang, "Metamodeling for high dimensional simulation-based design problems," *Journal of Mechanical Design,* vol. 132, p. 051009, 2010.

[6] Y. Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, 2001, pp. 1101-1108.

[7] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, 2008, pp. 1663-1670.

[8] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, 2009, pp. 983-989.

[9] H. K. Singh and T. Ray, "Divide and Conquer in Coevolution: A Difficult Balancing Act," in *Agent-Based Evolutionary Search*, ed: Springer, 2010, pp. 117-138.

[10] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, 2010, pp. 1-8.

[11] 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*, ed: Springer, 2010, pp. 300-309.

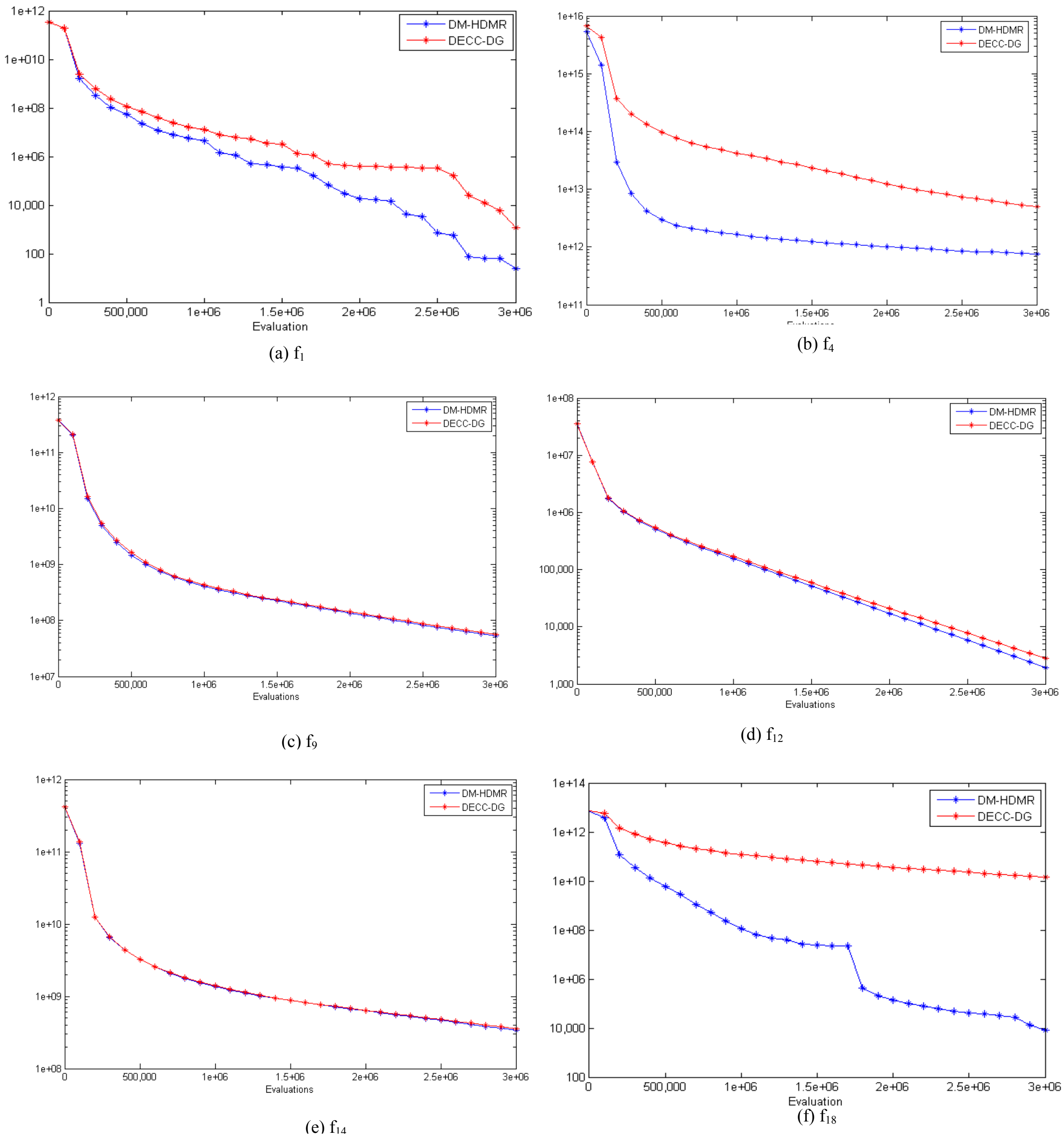
[12] E. Sayed, D. Essam, and R. Sarker, "Using hybrid dependency identification with a memetic algorithm for large scale optimization problems," in *Simulated Evolution and Learning*, ed: Springer, 2012, pp. 168-177.

[13] E. Sayed, D. Essam, and R. Sarker, "Dependency Identification technique for large scale optimization problems," in *Evolutionary Computation (CEC), 2012 IEEE Congress on*, 2012, pp. 1-8.

[14] H. Rabitz and Ö. F. Aliş, "General foundations of high - dimensional model representations," *Journal of Mathematical Chemistry,* vol. 25, pp. 197-233, 1999.

[15] I. Sobol, "Theorems and examples on high dimensional model representation," *Reliability Engineering & System Safety,* vol. 79, pp. 187-193, 2003.

[16] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," *Nature Inspired Computation and Applications Laboratory, USTC, China,* 2009.



(a) f_1 (b) f_4 (c) f_9 (d) f_{12} (e) f_{14} (f) f_{18}

Fig. 2. Convergence plots of f_1, f_4, f_9, f_{12}, f_{14} and f_{18}. Each point on the graph is the average over 25 independent runs.