A problem decomposition approach for large-scale global optimization problems

A problem decomposition approach for large-scale global optimization problems

A V Vakhnin[1], E A Sopov[1,2], I A Panfilov[1,2], A S Polyakova[1] and D V Kustov[2]

[1] Reshetnev Siberian State University of Science and Technology, 31, Krasnoyarskiy Rabochiy av., Krasnoyarsk, 660037, Russia
[2] Siberian Federal University, 79 Svobodny pr., 660041 Krasnoyarsk, Russia

E-mail: alexeyvah@gmail.com, evgenysopov@gmail.com, crook_80@mail.ru, polyakova_nasty@mail.ru, kustov.dv@gmail.com

Abstract. In fact, many modern real-world optimization problems have the great number of variables (more than 1000), which values should be optimized. These problems have been titled as large-scale global optimization (LSGO) problems. Typical LSGO problems can be formulated as the global optimization of a continuous objective function presented by a computational model of «Black-Box» (BB) type. For the BB optimization problem one can request only input and output values. LSGO problems are the challenge for the majority of evolutionary and metaheuristic algorithms. In this paper, we have described details on a new DECC-RAG algorithm based on a random adaptive grouping (RAG) algorithm for the cooperative coevolution framework and the well-known SaNSDE algorithm. We have tuned the number of subcomponents for RAG algorithm and have demonstrated that the proposed DECC-RAG algorithm outperforms some state-of-the-art algorithms with benchmark problems taken from the IEEE CEC'2010 and CEC'2013 competitions on LSGO.

1. Introduction

Today, there are a lot of relevant real-world optimization problems that involve many variables into optimization, for example [1]–[6]. These optimization problems with high dimensionality are known as large-scale global optimization problems (LSGO). LSGO problems are especially difficult because of the following important factors. Firstly, the search space of an optimization problem grows exponentially as the number of decision variables increases. This effect is known as the curse of dimensionality. Secondly, the type of the problem is the «Black-Box» (BB) optimization. We have no information about properties of the objective function landscape. Thirdly, the fitness evaluation of a solution for large-scale problems is usually computationally expensive and the number of evaluations is limited.

Without loss of generality, a BB LSGO problem can be stated as follows [7]:

$$f(\bar{x}) = f(x_1, x_2, \dots, x_n) \to \underset{\bar{x}\in X}{\min/\max} \qquad (1)$$

$$x_i^L \le x_i \le x_i^U, i = \overline{1,n} \qquad (2)$$

$$g_j(x_1, x_2, \dots, x_n) \le 0, j = \overline{1,m} \quad (3)$$

$$h_k(x_1, x_2, \dots, x_n) = 0, k = \overline{1,l} \quad (4)$$

where $\bar{x} \in X$, $X \subseteq R^n$ denotes the continuous decision space, $\bar{x} = (x_1, x_2, \dots, x_n) \in R^n$ is a real-value vector of decision variables. $f : X \to R^1$ is a real-value continuous nonlinear objective function. In (2), x_i^L and x_i^U define lower and upper side constrains for search interval, respectively. In this case, (3) and (4) define inequality and equality constraints, respectively. In this study, we consider the unconstrained LSGO minimization problem.

As it is known, metaheuristics show good performance for solving LSGO problems [8]. One of the effective LSGO technique applies methods based on cooperative coevolution (CC) framework [9]. The general idea of CC is connected with dividing a large optimization problem into several subcomponents and optimize them independently in order to solve the large optimization problem. In our previous papers [10], [11], we have proposed a novel variable grouping algorithm for CC framework that was titled as «Random Adaptive Grouping» (RAG). We have combined the well-known SaNSDE algorithm and RAG with CC framework. The whole metaheuristic algorithm is titled as DECC-RAG.

In this study, we have demonstrated the results of the performance investigation for DECC-RAG with different numbers of subcomponents on the IEEE LSGO CEC'10 and CEC'13 benchmarks. We have performed the detailed analysis of the DECC-RAG using statistics methods and Wilcoxon signed-rank test. It can be concluded that the proposed DECC-RAG algorithm outperforms some well-known state-of-the-arts algorithms on the LSGO CEC'10 and CEC'13.

The rest of the paper is organized as follows: Section II gives the preliminaries; Section III describes the proposed DECC-RAG algorithm; Section IV contains the descriptions of numerical experiments and discussed results; Section V concludes this paper and discussed further research.

2. Preliminaries

2.1. Classical Differential Evolution (DE) and Self-adaptive Differential Evolution with Neighborhood Search (SaNSDE)

Differential evolution (DE) is one the most popular and efficient evolutionary algorithm proposed for optimization in the space of real variables. DE is a stochastic, population-based search strategy developed by [12]. DE and its varieties [13]-[16] have good performance in on optimization problems of different difficulty levels. One of the further developments of DE is the SaNSDE algorithm proposed by [17]. We have chosen this algorithm for our investigation because of self-adaptive tuning of its parameters during optimization process.

As known, the performance of any evolutionary algorithm strongly depends on its control parameters. The general list of DE parameters contains the type of mutation, the scale factor value F and the crossover probability value CR. Frequently, $F \in [0; 2]$, $CR \in [0; 1]$. The main feature of the SaNSDE algorithm is that the algorithm stochastically selects a type of mutation and values of CR and F, and then adapts F and CR values based on the success of implementing a mutation operation. After a predefined number of generations, the SaNSDE recalculates probabilities for selection of a type of mutation and values of CR and F.

There exist many approaches for solving LSGO problems using DE and other evolutionary algorithms. We can divide all approaches into two main categories: cooperative coevolution (CC) algorithms with problem decomposition strategy and non-decomposition-based methods. As it has been shown in many studies, CC approaches usually demonstrates higher performance. The most popular CC approaches use different strategies for grouping of objective variables. Some well-known techniques are the static grouping [9], the random dynamic grouping [18] and the learning dynamic grouping [19].

2.2. Cooperative coevolution and variable grouping

Decomposition methods based on cooperative co-evolution are the most popular and widely used approaches for solving LSGO problems. Cooperative coevolution (CC) is an evolutionary framework that divides a solution vector of an optimization problem into several subcomponents and optimizes them independently in order to solve the optimization problem.

The first attempt to divide solution vectors into several subcomponents was proposed by [20]. The approach proposed by Potter and Jong (CCGA) decomposes a n-dimensional optimization problem into n one-dimensional problems (one for each variable). The CCGA employs CC framework and the standard GA. Potter and Jong had investigated two different modification of the CCGA: CCGA-1 and CCGA-2. The CCGA-1 evolves each variable of objective in a round-robin fashion using the current best values from the other variables of function. The CCGA-2 algorithm employs the method of random collaboration for calculating the fitness of an individual by integrating it with the randomly chosen members of other subcomponents. Potter and Jong had shown that CCGA-1 and CCGA-2 outperforms the standard GA.

The following pseudo-code presents general CC stages:

Pseudo-code of Cooperative Coevolution

```
Decompose objective vector into m smaller subcomponents;
while (termination condition is not achieved) do
  for i = 1 to m
    optimize i-th subcomponents with some EA;
  end for
end while
  return best found solution
```

The CC method is used for a wide range of real-world applications [21], [22] and [23].

3. Proposed Approach

We have analyzed pros and cons of grouping-based methods and DE-based approached, and have proposed a new EA for solving large-scale global optimization problems. The main idea of the proposed search algorithm is to combine of an original method of grouping variables for the CC with problem decomposition strategy with the self-adaptive DE (SaNSDE). The choice of the self-adaptive approach is necessary as we have no any information on a dependence between variables. Thus, parameters of the search algorithms should be adapted during the optimization process as information about the grouping quality becomes available.

As it is known, the CC approach can be efficient only if the grouping of variables is correct. As shown in [19], the learning dynamic grouping is not able to divide variables into correct subcomponents for many LSGO problems.

In the proposed approach, the grouping of variables is random and adaptive. In the approach, the number of grouped variables is equal for each subcomponent. Such limitation excludes the following problems:

- uneven distribution of computational resources between search algorithms (population sizes of EAs for each subcomponent);
- tuning minimum and maximum numbers of variables into group.

The proposed method of grouping (RAG (random adaptive grouping)) works as follows. The n-dimensional solution vector is divided into m s-dimensional sub-components (m x s = n). We randomly group variables into groups of equal sizes using the uniform distribution. As we need to estimate the quality of the distribution of variables, we will perform the EA run within the predefined budget T of the fitness function evaluation (each EA optimizes its corresponding subcomponent). During the optimization period of the algorithm, we record the increment of the function in each subcomponent Δf_i. After that, we will choose m/2 subcomponents with the worse performance (smallest Δf_i) and randomly mix indices of its variables. Finally, we will reset all EA parameters for the worst m/2 sub-components after regrouping variables and reset every Δf_i values. The reset is necessary because of the fact that new grouping of variables defines a completely different optimization problem.

The complete algorithm is called DECC-RAG. The procedure of DECC-RAG can be descripted by the following pseudo-code:

Pseudo-code of DECC-RAG algorithm

```
Set FEV_global, T, m, FEV_local = 0;
An n-dimensional object vector is randomly divided into m s-dimensional subcomponents;
Randomly mix indices of variables;
while (FEV > 0) do
  for i=1 to m
    Evolve the i-th subcomponent with SaNSDE algorithm, record CBS and PBS;
    $\Delta f_i$+=|PBS*-CBS**|
  end for
  if (FEV_local >= T)
   then choose m/2 subcomponents with the worse performance (m/2 smallest Δfi) and randomly mix indices of its subcomponents, restart parameters of SaNSDE in these m/2 subcomponents, FEV_local = 0, reset $\Delta f_i$ values;
  end if
end while
return the best found solution.
```

**previous best found solution*
***current best found solution*

4. Experimental Settings and Results

We have evaluated the performance of DE, SaNSDE and the proposed DECC-RAG algorithm with different group size on the 20 LSGO benchmark problems provided within the CEC'2010 special session on Large Scale Global Optimization [24] and on the 15 LSGO benchmark problems provided within the CEC'13 special session on Large Scale Global Optimization [25]. These benchmark problems have been specially endowed with the properties that real-world problems have.

The DECC-RAG algorithm settings are the next: NP = 50 (population size for each subcomponent) and T = 3x105. T is a parameter that represents a number of FEVs (function evaluations) before the stage of randomly mixing of the worse m/2 subcomponents.

All experimental settings are as proposed in the rules of the CEC'2010 and CEC'2013 LSGO competition were used for experiments:

- dimensions for all problem are D = 1000;
- 25 independent runs for each benchmark problem;

- 3x106 fitness evaluations in each independent run of algorithm;
- number of subcomponents m is {4, 8, 10, 20, 40, 50, 100}. We use the following notation: DECC-RAG(m);
- the performance of algorithms is estimated using the median value of the best found solutions.

All experiments were executed on the following system: OS: Ubuntu 16.04 LTS, CPU: AMD Ryzen 7 1700x (3.4GHz), 16 threads, RAM: 16GB, IDE: Code::Blocks, Language: C++, Compiler: g++ (gcc) with O3 optimization flag.

As it is known, LSGO problems are computationally expensive. Table 1 and table 2 show the runtime of 10000 fitness evaluations for each benchmark problem using 1 thread of the AMD Ryzen 7 1700x CPU on LSGO CEC'10 and LSGO CEC'13, respectively. Note that, in this study, we have used gcc compiler with O3 optimization flag to reduce program code running time.

In this study, we have implemented DE, SaNSDE and our DECC-RAG algorithms using C++ language. Also, we have implemented all our numerical experiments using the OpenMP framework for parallel computing with 16 threads, where each thread was allocated for one benchmark problem.

Table 1. Runtime of 10000 FEs (in seconds) on the CEC'10 LSGO Benchmark Problems.

F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
0.467	0.234	0.25	0.543	0.258	0.276	0.025
F8	**F9**	**F10**	**F11**	**F12**	**F13**	**F14**
0.025	0.613	0.397	0.421	0.02	0.023	0.79
F15	**F16**	**F17**	**F18**	**F19**	**F20**	-
0.566	0.617	0.018	0.024	0.02	0.03	-

Table 2. Runtime of 10000 FEs (in seconds) on the CEC'13 LSGO Benchmark Problems.

F1	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**
2.11	2.63	2.66	2.20	2.77	2.87	0.7
F8	**F9**	**F10**	**F11**	**F12**	**F13**	**F14**
2.6	3.16	2.25	2.42	0.03	2.5	2.41
F15	-	-	-	-	-	-
1.92	-	-	-	-	-	-

Table 3 and table 4 show results of Wilcoxon rank-sum test of statistical significance in the results of 25 independent runs for DECC-RAG (8) vs other DECC-RAG(m), DE and SaNSDE. The calculation of p-values has been performed using the R language in the R-studio software. The p-value for all tests was equal to 0.05. A rank of algorithms is defined by the median value, smaller median value defines smaller rank. Figure 1 and figure 2 demonstrate average ranks of DE, SaNSDE and DECC-RAG(m), respectively. As can be noted in figures 1 and 2, on two benchmark sets, DECC-RAG with 8 subcomponents has high performance.

Table 3. Wilcoxon Rank-sum Test (significantly, $p < 0.05$) DECC-RAG(8) vs Other EAs on LSGO CEC'10 Problems.

vs DECC-RAG(8)		**LSGO CEC'10**	**vs DECC-RAG(8)**		**LSGO CEC'10**
	+ (better)	1		+ (better)	4
DE	- (worse)	18	DECC-RAG(20)	- (worse)	15
	≈ (no sig.)	1		≈ (no sig.)	1

vs DECC-RAG(8)		LSGO CEC'10	vs DECC-RAG(8)		LSGO CEC'10
SaNSDE	+ (better)	2	DECC-RAG(40)	+ (better)	2
	- (worse)	15		- (worse)	18
	≈ (no sig.)	3		≈ (no sig.)	0
DECC-RAG(4)	+ (better)	8	DECC-RAG(50)	+ (better)	1
	- (worse)	9		- (worse)	19
	≈ (no sig.)	3		≈ (no sig.)	0
DECC-RAG(10)	+ (better)	4	DECC-RAG(100)	+ (better)	0
	- (worse)	9		- (worse)	20
	≈ (no sig.)	7		≈ (no sig.)	0

Table 4. Wilcoxon Rank-sum Test (significantly, $p < 0.05$) DECC-RAG(8) vs Other EAs on LSGO CEC'13 Problems.

vs DECC-RAG(8)		LSGO CEC'13	vs DECC-RAG(8)		LSGO CEC'13
DE	+ (better)	0	DECC-RAG(20)	+ (better)	3
	- (worse)	14		- (worse)	11
	≈ (no sig.)	1		≈ (no sig.)	1
SaNSDE	+ (better)	2	DECC-RAG(40)	+ (better)	1
	- (worse)	11		- (worse)	13
	≈ (no sig.)	2		≈ (no sig.)	1
DECC-RAG(4)	+ (better)	8	DECC-RAG(50)	+ (better)	1
	- (worse)	5		- (worse)	13
	≈ (no sig.)	2		≈ (no sig.)	1
DECC-RAG(10)	+ (better)	3	DECC-RAG(100)	+ (better)	0
	- (worse)	6		- (worse)	14
	≈ (no sig.)	6		≈ (no sig.)	1

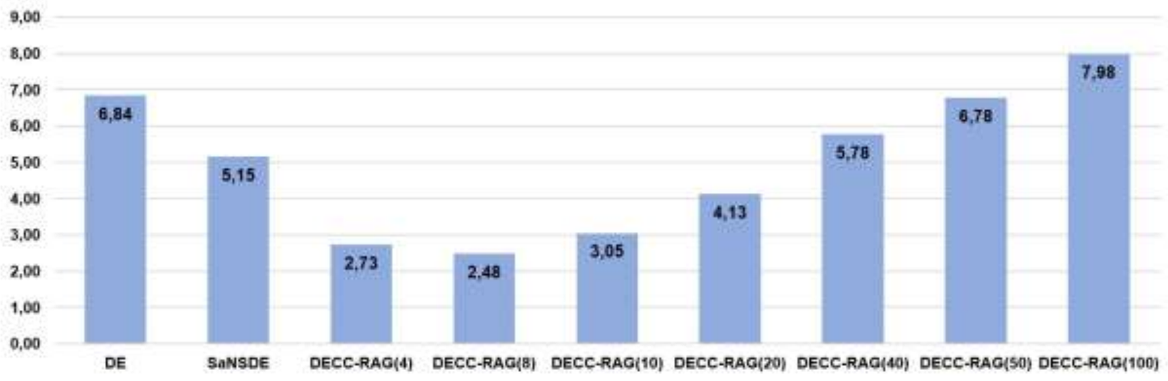

Figure 1. The DECC-RAG(m) ranking on the LSGO CEC'2010 benchmark problems.

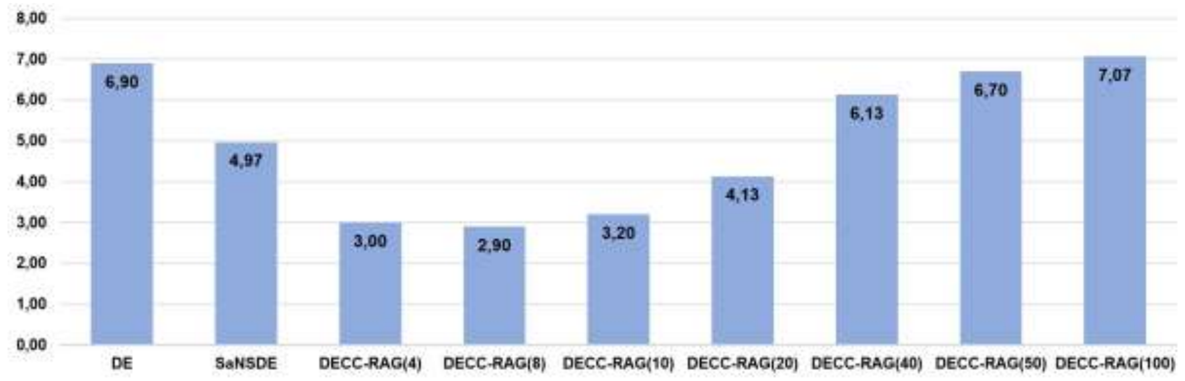

Figure 2. The DECC-RAG(m) ranking on the LSGO CEC'2013 benchmark problems.

We also have performed a comparison of DECC-RAG(8) vs other well-known state-of-the-art algorithms (DMS-L-PSO [26], DECC-G [18], MLCC [27], DECC-DG [19]) on the LSGO CEC'10 and CEC'13 benchmarks, respectively. The numerical results of the algorithms are taken from [28]. Figure 3 demonstrates average ranks of DECC-RAG(8) vs other well-known state-of-the-art evolutionary algorithms on the LSGO CEC'10 and CEC'13, respectively.

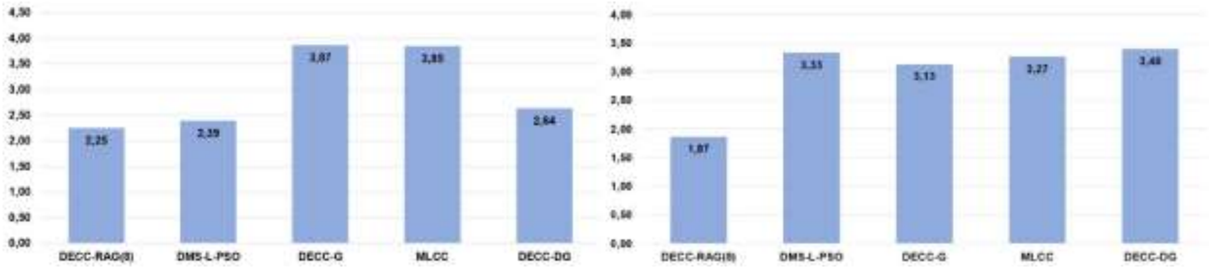

Figure 3. The DECC-RAG(8) and state-of-the-art algorithms ranking on the LSGO CEC 2010 and 2013.

5. Conclusions

In this study, we have proposed a new EA for large-scale global optimization problems and investigated its parameters. The approach uses an original random adaptive grouping method for cooperative coevolution framework.

The novelty of the proposed approach is based on including a feedback on success of random grouping. Saving good combinations of variables in subcomponents allows improving the decomposition stage for both separable and non-separable LSGO problems. The breakthrough of the approach is that the best experimental results are achieved for the small number of subcomponents with large number of variables. This means that the DECC-RAG provides an efficient problem decomposition, while the conventional methods needs to deal with groups with small number of variables (classical CCGA-1 and CCGA-2 uses groups with only 1 variable).

We have tested the proposed DECC-RAG algorithm on the representative set of 20 benchmark problems from the CEC'10 LSGO special session and competition and CEC'13 LSGO special session and competition, and have compared the results of the numerical experiments with other classic state-of-art techniques, such as DE and SaNSDE. We have estimated the performance of the DECC-RAG for different sizes of subcomponents, and can conclude that the best performance is obtained with the number of groups equal to 8 (m = 8).

The issues needed to be further studied are:

- designing more effective self-adaptive methods of grouping variables;
- improving the general performance of SaNSDE algorithm for LSGO problems.

In further work, we will provide more detailed analysis of the DECC-RAG performance depending on the number of individuals.

References

[1] Koh B I, Reinbolt J A, George A D, Haftka R T and Fregly B J 2009 Limitations of parallel global optimization for large-scale human movement problems *Med. Eng. Phys.* pp 515–21

[2] Wang Y, Li B and Weise T 2010 Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems *Inf. Sci. (Ny)* pp 2405–20

[3] Villaverde A F, Egea J A and Banga J R 2012 A cooperative strategy for parameter estimation in large scale systems biology models *BMC Syst. Biol.*

[4] Wang C and Gao J 2012 High-dimensional waveform inversion with cooperative coevolutionary differential evolution algorithm *IEEE Geosci. Remote Sens. Lett.*

[5] Mei Y, Li X and Yao X 2014 Variable neighborhood decomposition for Large Scale Capacitated Arc Routing Problem *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014* 1313–20

[6] Mei Y, Li X and Yao X 2014 Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems *IEEE Trans. Evol. Comput.* pp 435–49

[7] Vanderplaats G N 1999 *Numerical Optimization Techniques for Engineering Design - With Applications* (McGraw-Hill)

[8] Mahdavi S, Shiri M E and Rahnamayan S 2015 Metaheuristics in large-scale global continues optimization: A survey *Inf. Sci. (Ny).*

[9] Potter M A and Jong K A 2000 Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents *Evolutionary Computation* **8(1)** 1–29

[10] Vakhnin A and Sopov E 2018 A Novel Method for Grouping Variables in Cooperative Coevolution for Large-scale Global Optimization Problems *Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics* Vol 1 pp 261-8

[11] Sopov E and Vakhnin A 2018 An Investigation of Parameter Tuning in the Random Adaptive Grouping Algorithm for LSGO Problems *Proceedings of the 10th International Joint Conference on Computational Intelligence* Vol 1 pp 255-63

[12] Storn R and Price K 1995 *Differential Evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces* (International Computer Science Institute) pp 1–15

[13] Teo J 2006 Exploring dynamic self-adaptive populations in differential evolution *Soft Comput. A Fusion Found. Methodol. Applicat.* **10(8)** 673–86

[14] Fan H-Y and Lampinen J 2003 A trigonometric mutation operation to differential evolution *J. Global Optimization* **27(1)** 105–29

[15] Rahnamayan S, Tizhoosh H R and Salama M M A Opposition-based differential evolution *IEEE Trans. Evol. Comput.* **12(1)** 64–79

[16] Zhang Q Z and Sanderson A C 2009 JADE: Adaptive differential evolution with optional external archive *IEEE Trans. Evol. Comput.* **13(5)** 945–58

[17] Yang Z, Tang K and Yao X Self-adaptive differential evolution with neighborhood search *2008 IEEE Congress on Evolutionary Computation, CEC 2008* pp 1110–6

[18] Yang Z, Tang K and Yao X 2008 Large scale evolutionary optimization using cooperative coevolution *Information Sciences* **178(15)** 2985–99
[19] Omidvar M N, Li X, Mei Y and Yao X 2014 Cooperative co-evolution with differential grouping for large scale optimization *IEEE Trans. Evol. Comput.* **18(3)** 378–93
[20] Potter M A and Jong K A 1994 A cooperative coevolutionary approach to function optimization *Proc. of International Conference on Parallel Problem Solving from Nature* **2** 249–57
[21] Barrière O and Lutton E 2009 Experimental analysis of a variable size mono-population cooperative-coevolution strategy *Studies in Computational Intelligence* pp 139–52
[22] García-Pedrajas N, Hervás-Martínez C and Muñoz-Pérez J 2003 COVNET: A cooperative coevolutionary model for evolving artificial neural networks *IEEE Transactions on Neural Networks* **14(3)** pp 575–96
[23] Liu H, Yao X Zhao Q and Higuchi T 2001 Scaling up fast evolutionary programming with cooperative coevolution *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE)* **2** pp 1101–8
[24] Ke T, Xiaodong L, Zhenyu Y and Thomas W 2010 *Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization* (Tech. report, Univ. Sci. Technol. China) pp 1–21
[25] Li X, Tang K, Omidvar M N, Yang Z and Qin K 2013 *Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization* (Technical Report, Evolutionary Computation and Machine Learning Group, RMIT University, Australia)
[26] Liang J J and Suganthan P N 2005 Dynamic multi-swarm particle swarm optimizer *Proceedings - 2005 IEEE Swarm Intelligence Symposium, SIS 2005* pp 522–8
[27] Yang Z, Tang K and Yao X 2008 Multilevel cooperative coevolution for large scale optimization *2008 IEEE Congress on Evolutionary Computation, CEC 2008* pp 1663–70
[28] Yang Q, Chen W-N, Deng J D, Li Y, Gu T and Zhang J A level-based learning swarm optimizer for large-scale optimization *IEEE Trans. Evol. Comput.* **22(4)** 578–94