Improved random adaptive grouping approach for solving unconstrained LSGO problems

A Vakhnin¹, E Sopov¹,² and I Panfilov¹,²

¹ Reshetnev Siberian State University of Science and Technology, 31, Krasnoyarsky Rabochny Av., Krasnoyarsk, 660037, Russian Federation
² Siberian Federal University, 79, Svobodny Av., Krasnoyarsk, 660041, Russian Federation

E-mail: alexeyvah@gmail.com

Abstract. Large-scale global optimization (LSGO) problems are emergent in many domains of applied sciences. LSGO is a hard challenge for the majority of state-of-the-art optimization methods. Evolution algorithms (EAs) combined with Cooperative Coevolution (CC) are able to perform well when solving many real-world LSGO problems. The grouping of variables at the problem decomposition stage has a significant impact on the performance of the CC approach. This study proposes an improvement of the previously developed random adaptive grouping (RAG) approach for CC. The new method is titled as RAG2, and the whole optimization algorithm, based on RAG2, is called CC-SHADE-RAG2. The influence of the choice of the population size and the number of subcomponents on the algorithm performance have been investigated using the IEEE LSGO CEC’2013 benchmark. The set of test problems in the benchmark contains fifteen functions with dimensionality equal to one thousand. We have also compared the performance of the novel algorithm with some EAs, which are applied for solving LSGO problems.

1. Introduction

In general, actual optimization tasks are viewed as «Black-Box» (BB) continuous optimization problem (knowledge about the optimized function is not available) [1]. A BB optimization problem is formulated as to:

\[
\text{minimize } f(\bar{x}) = f(x_1, x_2, \ldots, x_N) \\
\text{such that } x_i^L \leq x_i \leq x_i^R, \ i = 1, N
\]  

where (1) \( \bar{x} \) is objective vector with real-valued numbers, \( f(\bar{x}): R^N \rightarrow R \) is (fitness) function with \( N \) variables. In (2), \( x_i^L \) and \( x_i^R \) mean left and right borders for search interval, respectively. Unconstrained optimization problems have been considered in this paper.

Also, these contemporary challenges have many real-valued objective variables, tasks of this type are called large-scale global optimization (LSGO) problems [2]. LSGO is a new branch in optimization area. In the last decade, the interest of thousands of scientists in the LSGO field can be observed. Classic evolution algorithms (EAs) and their metaheuristics show better performance than classic optimization techniques in solving BB problems [3]. However, effectiveness of EAs significantly is reduced in solving problems with a high number of variables (hundreds or thousands). This is due to the fact that
the search space, which has box constraints, increases exponentially as the number of variables grows. This phenomenon is widely known in the scientific literature and it is called the Curse of dimensionality (CoD).

For today, Cooperative coevolution (CC) is one of the most effective way of reducing the negative influence of the CoD. CC, first proposed by Potter and De Jong in 1994 [4], is a framework for EAs that divides the vector of optimization problem into several parts (subcomponents) and optimizes them independently. The grouping of variables is strongly effects on the CC performance. To achieve better solution value, EA has to optimize some variables at the same time (variables must be in the same subcomponent). There are many different variable grouping methods, for example: [5], [6], [7] and [8].

In this article, an improved approach for variable grouping, which is named «Random Adaptive Grouping version 2» (RAG2), has been proposed. RAG2 is based on our previously proposed grouping method «Random Adaptive Grouping version 1» (RAG) [9]. CC-SHADE-RAG2 is our proposed algorithm for solving LSGO problems which is based on CC with RAG2 and effective self-adaptive SHADE algorithm.

The rest of the paper is organized as follows. Section 2 describes related work. In Section 3, the Random Adaptive Grouping 2 approach and CC-SHADE-RAG2 algorithm are described. In Section 4, the experimental setup and results of numerical experiments are presented and discussed. In the conclusion, the results and further research are discussed.

2. Related Work

2.1. Classical Differential Evolution (DE) and Success-History based Adaptive DE (SHADE)

Differential evolution (DE), first proposed by Storn and Price [10], is one of the useful instruments for BB optimization with continuous variables. To date, there are a lot of varieties of classic DE, such as SaDE [11], JADE [12], and SaNSDE [13] demonstrate good performance in solving BB optimization problems. One of the successful DE improvements is the SHADE algorithm [14]. SHADE has the following features, it self-adapts values of $F$ and $CR$ parameters using historical memory. Also, SHADE has an external archive. This external archive consists of replaced individuals to keep the memory of the best previous solutions and uses them to generate new solutions.

2.2. Cooperative Coevolution (CC) framework

CC is an effective framework for EAs to solve LSGO problems [15]. CC uses EA as an optimization core to optimize each subcomponent. CC framework decomposes solution vector into shorter vectors and optimizes them independently via some EA. Figure 1 demonstrates the scheme of CC with main steps.

![Figure 1](image1.png)

**Figure 1.** The general scheme of Cooperative Coevolution framework.
The original CC approach has been proposed by Potter and Jong [4], it is been referred to CCGA. CCGA consists of CC framework and classic genetic algorithm (GA). Authors have studied two following modifications of the CCGA: CCGA-1 and CCGA-2. These two algorithms have used different methods collaboration of subcomponents. The CCGA-1 optimizes each function variable in a round-robin fashion using the best records from the other attributes of function. The CCGA-2 algorithm uses the method of stochastic collaboration for counting the fitness of an individual by integrating it with the randomly taken members of other subcomponents. In [4], Potter and Jong have shown that CCGA-1 and CCGA-2 outperform the standard GA. It is important to note that CC framework is efficiently used for a large spectrum of real-world applications [16, 17] and [18].

3. Improved Random Adaptive Grouping

3.1. Original Random Adaptive Grouping (RAG)

The main concept of authentic RAG is recording the target function increments per subcomponent \((\Delta f_k), k = 1, m\) inward the learning period \((T)\) limits. \(m\) is the total sum of all sub-components. The learning period is the number of calculated functions that is needed to evaluate EA performance at each subcomponent. Usually, we set \(T\) is equal to 10% of the total fitness budget \((0.1 \cdot FEVs)\). After the learning period, RAG randomly mixes variables in half subcomponents \((m/2)\) where the optimizer (EA) has demonstrated worse performance (based on \((\Delta f_k)\)). Thus, RAG saves subcomponents with good groupings of variables and randomly mixes poorly grouped variables. RAG resets self-adapted EA parameters in bad groups. If the calculation budget is not over, then the optimization procedure will be continued. If the stopping condition is reached, then the evolutionary method returns the best-found solution.

3.2. Random Adaptive Grouping version 2 (RAG2)

This subsection describes the RAG2 approach. Based on the knowledge from the article [15], optimization CC techniques can be split into three categories: the static [19], the random dynamic [20] and the learning dynamic [21]. The static methods can be applied if the connection type between variables is known. It is risky to use this type of methods, because the optimization tasks are frequently BB type. By contrast, the learning dynamic methods spend many FEVs before the main optimization process to find the interaction between variables and then group them into subcomponents. Also, the learning dynamic methods do not guarantee correct groupings. Thus, to save the fitness budget, it is necessary to obtain the useful information about grouping variables during the optimization process.

Our RAG2 method uses \(\bar{x}_{\text{Increment}}\) vector to record the coordinate incrementations for each variable between new individual and replaced individual using the modified selection DE operator (3),

\[
\bar{x}_{\text{Increment}}^j = \{(u_{j,G} - x_{j,G}) \mid f(u_{j,G}) \leq f(x_{j,G})\}
\]

(3)

where \(j\) is the index of individual in population. \(G\) is the generation number. \(u_{j,G}\) is the mutant vector which is produced by the mutation operator. \(x_{j,G}\) — current solution of \(j\)-th individual. If the fitness value of an individual has improved, then it is necessary to calculate the modular difference between the new and old coordinates. This \(\bar{x}_{\text{Increment}}\) vector accumulates information during the learning period about how much each coordinate has been changed during the optimization process. After the learning period is achieved, RAG2 sorts objective variables according values in \(\bar{x}_{\text{Increment}}\) vector (from greater to lesser values). This type of grouping adds variables that needed a larger increment to improve the fitness function into the first subcomponents. Variables that needed a small increment are grouped into the last subcomponents. In the last stage of RAG2 algorithm, we have added randomness to this type of variables grouping in order to avoid the algorithm getting into local optima. RAG2 randomly mixes variables in neighboring subcomponents using the following rule (4). Before starting a new learning period, EA has to zero the values of the \(\bar{x}_{\text{Increment}}\) vector.
randomly mix variables in subcomponent$_k$ and subcomponent$_{k+1}, k = 1, m, k = k + 2$ (4)

The algorithm title is CC-SHADE-RAG2. The procedure of CC-SHADE-RAG2 is shown using the following pseudo-code in Figure 2.

Pseudo-code of CC-SHADE-RAG2 algorithm

Set FEV, 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
    Optimize the i-th subcomponent with SHADE algorithm with condition (3);
  end for
  if (FEV_local >= T) // start RAG2
    Sort variables ascending according to $X_{increment}$ vector;
    Apply (4) rule to randomly mix variables in neighboring groups;
    Zero all $X_{increment}$ vector values;
    FEV_local = 0;
  end if // end RAG2
end while
return the best-found solution.

Figure 2. Pseudo-code of CC-SHADE-RAG2 algorithm.

4. Experimental setup and discussion numerical experiments

4.1. Experimental setup

In this article, the following PC system has been used to execute all numerical experiments. There were 4 CPUs (one Ryzen 7 1700x and three Ryzen 7 2700) to parallelize numerical experiments via MPICH2 (Message Passing Interface Chameleon). Each computer has 8 GB of RAM. The program code of CC-SHADE-RAG2 had realized on the C++ program language with GNU Compiler Collection (GCC). Operating system is Ubuntu 18.04 LTS.

The CC-SHADE-RAG2 performance has been evaluated with different sets of sub-components number and different population size on the fifteen artificial LSGO benchmark problems. The special session of CEC’2013 LSGO has introduced these problems [22] to evaluate various optimization techniques. Problems from this set have similar properties as real-world problems. The CC-SHADE-RAG2 has the following settings. $NP = \{25, 50, 100, 150, 200, 250, 300\}$ (size of population in each subcomponent), $m = \{4, 6, 8, 10\}$ (the sub-components number), the size of external archive is two more than population size. The size of historical memory ($H$) is 6. $T = 3 \cdot 10^5$ FEVs (learning period). EA can evaluate only $3 \cdot 10^6$ FEVs for each independent run. The statistic of the CC-SHADE-RAG2 has been collected in 25 autonomous runs. The performance of CC-SHADE-RAG2 has been compared with other effective state-of-the-art algorithms: MPS [23], SGCC [24], CC-RDG3 [25] and DGSC [26].

4.2. Numerical experiments and results

This part of section 4 demonstrates results of numerical experiments. Table 1 contains CC-SHADE-RAG2($m/pop\_size$) performance. The numbers in brackets mean algorithm parameters. $m$ is the number of subcomponents CC framework, $pop\_size$ is the number of individuals in each subcomponent. Table 1 shows results with the best number of $NP$ for 4, 6, 8 and 10 subcomponents. The first column shows the number of benchmark problem. Each cell contains mean value which was obtained from 25 independent runs of proposed CC-SHADE-RAG2. According to the Competition Congress of LSGO CEC’2013, to compare algorithms with each other, optimization algorithms get points for each benchmark problem. The 1st place is 25 points, the 2nd is 18, the 3rd is 15, the 4th is 12, the 5th is 10, the 6th is 8 and the 7th is 6. This rating system is used in Formula 1. The algorithm that gets the most points is total winner. As we can see from Table 1, the CC-SHADE-RAG2 combination with 4 subcomponents and 250 individuals has got the highest score (290).
Table 2 includes results of the CC-SHADE-RAG2 best parameter combinations of using Mann–Whitney U test ($p$-value is equal to 0.01). The first column and the first row provide designations for set of parameters. Each cell provides the values which have been calculated as follows. If one set of parameters (from column) has better, worse or equal performance to another (from row) on the benchmark problems, then add one score to this value, respectively. Each cell has ($w/l/e$) format, where $w$ and $l$ mean the total number of problems where column algorithm outperforms/underperforms row algorithm, respectively. $e$ means the total number of benchmark problems with the same performance of algorithms. The last column provides calculated value which are the score sum of row. Analyzing the data from Table 2, we can conclude that the CC-SHADE-RAG2 performance with following parameter sets (4/250), (6/150), (8/100) and (10/100) are very similar.

| Function | 4/250 | 6/150 | 8/100 | 10/100 | Total Score |
|----------|-------|-------|-------|--------|-------------|
| F1       | 2.15E-20 | 2.73E-21 | 7.14E-22 | 1.43E-22 |             |
| F2       | 2.02E+03 | 1.78E+03 | 1.76E+03 | 1.29E+03 |             |
| F3       | 2.11E+01 | 2.10E+01 | 2.09E+01 | 2.10E+01 |             |
| F4       | 2.13E+09 | 2.37E+09 | 2.66E+09 | 4.62E+09 |             |
| F5       | 9.88E+05 | 1.60E+06 | 2.18E+06 | 2.58E+06 |             |
| F6       | 1.06E+06 | 1.06E+06 | 1.06E+06 | 1.06E+06 |             |
| F7       | 2.62E+05 | 2.12E+05 | 2.22E+05 | 1.80E+05 |             |
| F8       | 6.59E+13 | 7.62E+13 | 8.60E+13 | 1.01E+14 |             |
| F9       | 9.20E+07 | 1.27E+08 | 1.47E+08 | 1.79E+08 |             |
| F10      | 9.38E+07 | 9.34E+07 | 9.31E+07 | 9.26E+07 |             |
| F11      | 8.59E+06 | 1.54E+07 | 2.36E+07 | 3.26E+07 |             |
| F12      | 1.37E+03 | 1.37E+03 | 1.38E+03 | 1.30E+03 |             |
| F13      | 9.67E+06 | 1.10E+07 | 1.28E+07 | 2.48E+07 |             |
| F14      | 1.67E+07 | 1.77E+07 | 2.17E+07 | 2.81E+07 |             |
| F15      | 7.95E+05 | 5.39E+05 | 4.05E+05 | 3.92E+05 |             |
| Total Score | 290 | 265 | 254 | 277 |             |

Table 1. CC-SHADE-RAG2($m$/$pop$$_size$) performance.

| (m/$pop$$_size$) | 4/250 | 6/150 | 8/100 | 10/100 | Total sum |
|------------------|-------|-------|-------|--------|-----------|
| 4/250            | -     | 2/5/8 | 4/6/5 | 6/7/2  | 12/18/15 |
| 6/150            | 5/2/8 | -     | 3/4/8 | 6/5/4  | 14/11/20 |
| 8/100            | 6/4/5 | 4/3/8 | -     | 4/4/7  | 14/11/20 |
| 10/100           | 7/6/2 | 5/6/4 | 4/4/7 | -      | 16/16/13 |

We have used (4/250) parameter set of CC-SHADE-RAG2 for comparison with other state-of-the-art EAs due to these parameters have the highest total score in Table 1. The structure of Table 3 is the same to Table 1. As we can see, CC-SHADE-RAG2 has been placed the second place.
Table 3. The CC-SHADE-RAG2 performance vs other state-of-the-art EAs on the LSGO CEC’2013.

| Function | CC-SHADE-RAG2 (4/250) | MPS | SGCC | CC-RDG3 | DGSC |
|----------|------------------------|-----|------|---------|------|
| F1       | 2.15E-20               | 6.68E+08 | 1.85E+03 | 1.14E-18 | 2.60E-04 |
| F2       | 2.02E+03               | 4.20E+03 | 8.94E+03 | 2.31E+03 | 7.15E+02 |
| F3       | 2.11E+01               | 1.94E+00 | 2.15E+01 | 2.04E+01 | 2.07E+01 |
| F4       | 2.13E+09               | 1.07E+11 | 1.23E+09 | 4.29E+04 | 3.77E+08 |
| F5       | 9.88E+05               | 1.20E+06 | 5.02E+06 | 2.04E+06 | 3.27E+06 |
| F6       | 1.06E+06               | 6.01E+03 | 1.06E+06 | 1.00E+06 | 1.06E+06 |
| F7       | 2.62E+05               | 7.19E+07 | 8.94E+03 | 2.31E+03 | 7.15E+02 |
| F8       | 6.59E+13               | 2.04E+14 | 1.54E+11 | 7.11E+03 | 1.85E+13 |
| F9       | 9.20E+07               | 1.66E+08 | 4.06E+08 | 1.57E+08 | 1.79E+08 |
| F10      | 9.38E+07               | 3.53E+06 | 9.38E+07 | 9.16E+07 | 9.38E+07 |
| F11      | 8.59E+06               | 2.20E+09 | 2.55E+07 | 2.18E+13 | 6.92E+09 |
| F12      | 1.37E+03               | 1.75E+04 | 3.56E+03 | 7.00E+02 | 2.93E+03 |
| F13      | 9.67E+06               | 9.87E+08 | 1.21E+07 | 6.43E+04 | 8.63E+08 |
| F14      | 1.67E+07               | 1.03E+09 | 2.07E+07 | 1.65E+09 | 1.32E+08 |
| F15      | 7.95E+05               | 2.76E+07 | 1.30E+06 | 2.30E+06 | 2.67E+07 |
| Total score | 278 | 214 | 211 | 295 | 218 |

5. Conclusions
This paper proposes improvement version of RAG method for cooperative coevolution framework. We have proposed novel grouping method RAG2 and CC-SHADE-RAG2 algorithm. This study had evaluated the CC-SHADE-RAG2 performance for LSGO CEC’2013 with different sets of parameters. In further research, CC-SHADE-RAG2 performance will be improved via using self-adaptation of mutation strategy.

References
[1] Muñoz M A, Sun Y, Kirley M and Halgamuge S K 2015 Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges *Inf. Sci.* **317** 224–45 doi:10.1016/j.ins.2015.05.010
[2] Mahdavi S, Rahnamayan S and Shiri M E 2016 Multilevel framework for large-scale global optimization *Soft Comp.* **21**(14) 4111–40 doi:10.1007/s00500-016-2060-y
[3] Anile A M, Cutello V, Nicosia G, Rascuna R and Spinella S 2005 Comparison among evolutionary algorithms and classical optimization methods for circuit design problems *2005 IEEE Cong. on Evol. Comp.* 1 765–72 doi:10.1109/CEC.2005.1554760
[4] Potter M A and Jong K A 1994 A cooperative coevolutionary approach to function optimization *Lect. Notes in Comp. Sci.* 249–57 doi:10.1007/3-540-58484-6_269
[5] Zhenyu Y, Ke T and Xin Y 2008 Multilevel cooperative coevolution for large scale optimization *2008 IEEE Cong. on Evol. Comp.* doi:10.1109/cecc.2008.4631014
[6] Chen W, Weise T, Yang Z and Tang K 2010 Large-Scale Global Optimization Using Cooperative Coevolution with Variable Interaction Learning *Parallel Problem Solving from Nature PPSN XI* pp 300–309 doi:10.1007/978-3-642-15871-1_31
[7] Mei Y, Li X and Yao X 2014 Cooperative Coevolution With Route Distance Grouping for Large-Scale Capacitated Arc Routing Problems *IEEE Trans. on Evol. Comp.* **18**(3) 435–49
Omidvar M N, Li X, Mei Y and Yao X 2014 Cooperative Co-Evolution With Differential Grouping for Large Scale Optimization IEEE Trans. on Evol. Comp. 18(3) 378–93 doi:10.1109/tevc.2013.2281503

[8] Omidvar M N, Li X, Mei Y and Yao X 2014 Cooperative Co-Evolution With Differential Grouping for Large Scale Optimization IEEE Trans. on Evol. Comp. 18(3) 378–93 doi:10.1109/tevc.2013.2281543

[9] Sopov E, Vakhnin A and Semenkin E 2018 On Tuning Group Sizes in the Random Adaptive Grouping Algorithm for Large-scale Global Optimization Problems Int. Conf. on App. Math. & Comp. Sci. (ICAMCS.NET) pp 134-1341 doi:10.1109/ICAMCS.NET46018.2018.00031

[10] Storn R and Price K 1997 Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces Jour. of Global Opt. 11(4) 341–59 doi:10.1023/a:1008202821328

[11] Qin A K, Huang V L and Suganthan P N 2009 Differential Evolution Algorithm With Strategy Adaptation for Global Numerical Optimization IEEE Trans. on Evol. Comp. 13(2) 398–417 doi:10.1109/tevc.2008.927706

[12] Zhang J and Sanderson A C 2009 JADE: Adaptive Differential Evolution With Optional External Archive IEEE Trans. on Evol. Comp. 13(5) 945–58 doi:10.1109/tevc.2009.2014613

[13] Yang Z, Tang K and Yao X 2008 Self-adaptive differential evolution with neighborhood search 2008 IEEE Congr. on Evol. Comp. pp 1110–6 doi:10.1109/CEC.2008.4630935

[14] Tanabe R and Fukunaga A 2013 Success-history based parameter adaptation for Differential Evolution 2013 IEEE Congr. on Evol. Comp. doi:10.1109/cec.2013.6557555

[15] Mahdavi S, Shiri M E and Rahnamayan S 2015 Metaheuristics in large-scale global continues optimization: A survey Inf. Sci. 295 407–28 doi:10.1016/j.ins.2014.10.042

[16] Boonlong K 2014 Vibration-Based Damage Detection in Beams by Cooperative Coevolutionary Genetic Algorithm Adv. in Mech. Eng. doi:10.1155/2014/624949

[17] Wang C and Gao J 2010 A new differential evolution algorithm with Cooperative Coevolutionary selection operator for waveform inversion 2010 IEEE Int. Geosci. and Remote Sensing Symp. pp 668–90 doi:10.1109/igars.2010.5652032

[18] 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. 180(12) 2405–20 doi:10.1016/j.ins.2010.02.015

[19] Potter M A and Jong K A 2000 Cooperative Coevolution: An Architecture for Evolving Co-adapted Subcomponents Evol. Comp. 8(1) 1–29 doi:10.1162/106365600568086

[20] Yang Z, Tang K and Yao X 2008 Large scale evolutionary optimization using cooperative coevolution Inf. Sci. 178(15) 2985–99 doi:10.1016/j.ins.2008.02.017

[21] Omidvar M N, Li X and Yao X 2010 Cooperative Co-evolution with delta grouping for large scale non-separable function optimization IEEE Congr. on Evol. Comp. doi:10.1109/cec.2010.5585979

[22] 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 Tech. Rep., Evol. Comp. and Machine Learning Group, RMIT University, Australia

[23] Bolufe-Rohler A, Fiol-Gonzalez S and Chen S 2015 A minimum population search hybrid for large scale global optimization 2015 IEEE Congr. on Evol. Comp. (CEC) doi:10.1109/cec.2015.7257125

[24] Liu W, Zhou Y, Li B and Tang K 2019 Cooperative Co-evolution with Soft Grouping for Large Scale Global Optimization 2019 IEEE Congr. on Evol. Comp. pp 318–25 doi:10.1109/CEC.2019.8790053

[25] Sun Y, Li X, Ernst A and Omidvar M N 2019 Decomposition for Large-scale Optimization Problems with Overlapping Components 2019 IEEE Congr. on Evol. Comp. pp 326–33 doi:10.1109/CEC.2019.8790204

[26] Li L, Fang W, Wang Q and Sun J 2019 Differential Grouping with Spectral Clustering for Large Scale Global Optimization 2019 IEEE Congr. on Evol. Comp. pp 334-41 doi:10.1109/CEC.2019.8790056