# Using cooperative coevolution in large-scale black-box constraint satisfaction problems

*Aleksei* Vakhnin[1,2*], and *Zakhar* Novikov[1]

[1]Siberian Federal University, Institute of Space and Information Technology, Krasnoyarsk, Russia
[2]Reshetnev Siberian State University of Science and Technology, Institute of Informatics and Telecommunications, Krasnoyarsk, Russia

**Abstract.** Solving constrained large-scale global optimization problems poses a challenging task. In these problems with constraints, when the number of variables is measured in the thousands, when the constraints are presented in the form of a black box, and neither the size nor the configuration of the feasible region is known, it is very difficult to find at least one feasible solution. In general, such a problem of finding a feasible region is known as a constraint satisfaction problem. In this paper, we have extended a well-known benchmark set based on constrained optimization problems up to 1000 variables. We have evaluated the CC-SHADE performance, to tackle constraints in large-scale search space. CC-SHADE merges the power of cooperative coevolution and self-adaptive differential evolution. Our extensive experimental evaluations on a range of benchmark problems demonstrate the strong dependence of the performance of CC-SHADE on the number of individuals and the subcomponent number. The numerical results emphasize the importance of using a cooperative coevolution framework for evolutionary-based approaches compared to conventional methods. All numerical experiments are proven by the Wilcoxon test.

## 1 Introduction

Solving complex, constrained global optimization problems is a comprehensive challenge across various domains, from engineering to finance [1, 2]. Such problems involve finding the best solution within a large search space while adhering to specific constraints or limitations. In real-world applications, these optimization tasks often appear in complex forms, making them not only computationally intensive but also demanding in terms of solution quality.

In general, constrained large-scale global optimization can be defined as the following:

$$Minimize: f(X), X = (x_1, x_2, \dots, x_n) \; and \; X \in S \quad (1)$$

$$Subject\ to: \begin{cases} g_i(X) \le 0, i = 1, \dots, p \\ h_j(X) = 0, j = p+1, \dots, m \end{cases} \quad (2)$$

* Corresponding author: alexeyvah@gmail.com

In general, we can transform equality (2) constraints into inequalities based on the equation (3):

$$\left|h_j(X)\right| - \varepsilon \leq 0, j = p+1,\ldots,m \tag{3}$$

A solution $X$ $is$ regarded as a feasible if $g_i(X) \leq 0, for \;\; i = 1,\ldots,p,$ and $\left|h_j(X)\right| - \varepsilon \leq 0,$ for $j = p+1,\ldots,m$. In this study, $\varepsilon \; = \; 0.0001$.

Traditional optimization algorithms often struggle to tackle these large-scale constrained problems effectively, mostly because of the curse of dimensionality. However, the using of cooperative coevolutionary-based evolutionary algorithms [3] presents a promising wide range of ways to address this challenge. Cooperative coevolution is one of the powerful optimization techniques used to solve large-scale global optimization problems by breaking them down into smaller subproblems. Large-scale optimization problems often suffer from high-dimensional search spaces, which can be computationally expensive to explore exhaustively. Cooperative coevolution decomposes the optimized vector of parameters into smaller, lower-dimensional subproblems. In the paper we handle only constraints, specifically, we try to find feasible solutions. Many approaches can be applied to improve solutions from the feasible region.

CSPs (constraint satisfaction problems), as indicated in references [4, 5], encompass a collection of problems wherein the conditions are required to adhere to a set of equality or inequality constraints. CSPs often present formidable complexity, necessitating a blend of heuristics and combinatorial search methods for timely resolution. Black-box constraint satisfaction problems (BB-CSPs) represent a critical domain in computer science and artificial intelligence, where the objective is to determine a valid assignment of values to a set of variables while respecting a set of constraints. The term "black-box" is indicative of the challenging nature of these problems, where the underlying constraint functions are often hidden or not directly accessible.

A collection of variables $X \; = \; \{X_1,\ldots,X_n\}$, each associated with discrete-valued domains $D \; = \; \{D_1,\ldots,D_n\}$, forms the basis of a CSP represented by a set of constraints $\mathrm{C} \; = \; \{\mathrm{C}_1,\ldots,\mathrm{C}_m\}$. Each constraint $\mathrm{C}_i$ is defined as a pair $(S_i, R_i)$, where $R_i$ is a relation $R_i \subseteq D_{S_i}$ established on a subset of variables $S_i \subseteq X$, referred to as the scope of $\mathrm{C}_i$. This relation encompasses all permissible tuples from $D_{S_i}$ according to the constraint. A solution is an assignment of values to variables $x \; = \; (x_1,\ldots,x_n)$, where $x_i \in D_i$, ensuring the satisfaction of each constraint. If a solution is attainable, the problem is considered satisfiable or consistent. The task of finding a solution for CSP is recognized as an NP-complete problem.

The paper is organized as follows. Section 2 outlines methodologies for addressing constrained black-box optimization problems and introduces the cooperative coevolution framework. Section 3 presents the fundamental concept of CC-SHADE for handling constraints within the extended benchmark set proposed in this study. In Section 4, the experimental setup and results are discussed, along with detailed information about the computation cluster. The concluding section summarizes the findings and outlines avenues for future research.

## 2 Related work

Black-box constrained optimization problems [7], where the objective function and constraints are not given in algorithmic form, have gained great attention in recent years. Various methods have been proposed to address the challenges posed by these types of problems. In this section, we review some of the key approaches that have been developed for solving black-box constrained optimization problems.

### 2.1 Evolutionary-based algorithms

Genetic algorithms (GA) are one of the most widely used evolutionary-based methods for constrained optimization. GAs maintain a population of candidate solutions and iteratively evolve them through selection, crossover, and mutation operations. In the papers [8, 9], we can find examples of using GA for solving CSPs.

Evolution Strategies (ES) are optimization algorithms that focus on parameter tuning and optimization in a black-box setting. ES maintains a population of candidate solutions and updates them through selection, recombination, and mutation operations. ES can be also applied to solving CSPs [10].

Differential Evolution (DE) is an evolutionary algorithm that operates by creating new candidate solutions based on the differences between randomly selected solutions in the population. DE has been adapted for handling constraints in solving CSPs [11, 12]. DE is known for its simplicity and robustness in handling various types of constraints.

Particle Swarm Optimization (PSO) is an optimization technique that mimics the social behavior of birds or fish. In the context of constrained optimization, particles in the swarm explore the solution space, and constraints are typically handled through penalty functions. Particles are guided toward feasible solutions through a combination of individual and global best-known solutions. Applications of PSO in solving CSPs can be found in [13, 14]

These evolutionary-based approaches offer a diverse set of tools for solving CSPs. The choice of method depends on the specific characteristics of the problem, the nature of constraints, and the available resources for optimization. Researchers and practitioners often experiment with different approaches to find the most suitable one for their particular problem [15].

### 2.2 Cooperative coevolution

Cooperative coevolution [16], a variant of the coevolutionary framework, is an optimization technique where a complex problem is divided into smaller, more manageable subcomponents. Each subcomponent corresponds to a part of the solution space and is evolved separately. By optimizing these subcomponents concurrently, a collective solution is gradually built, resulting in a more efficient exploration of the entire problem space.

In the context of constrained black-box optimization, the cooperative coevolution framework can be applied in two following ways. The first, a CC-based EA optimizes the objective function and constraints simultaneously. The seconds, a CC-based EA optimizes the constraints initially, it tries to find any feasible solution, and then the algorithm tries to optimize the objective function without violating the restrictions.

## 3 Proposed approach

In the scope of solving constrained black-box optimization problems, where the objective function and constraints are not explicitly provided, a CC-based EA has been investigated to efficiently tackle the complexities of such problems. In the previous study, we proposed and investigated an approach for solving LSGO (large-scale global optimization) problems [17]. The CC-SHADE approach is based on a combination of two components: cooperative coevolution [16] and SHADE [18]. By integrating these two techniques, our approach focuses on optimizing only the constraints, allowing us to find solutions in the complex landscape. The sort description of CC-SHADE is presented below.

The foundation of our approach lies in the cooperative coevolution framework, which breaks down the constrained optimization problem into manageable subcomponents. The primary objective of this component is to optimize a large vector of solution separately. We achieve this by decomposing the vector of the solution into smaller subproblems with smaller dimensions. The second important component of our approach is the use of a SHADE optimizer. SHADE has proven to be highly effective in optimizing complex, noisy, and black-box objective functions. In our approach, we employ SHADE to optimize only the constraints. SHADE is renowned for its self-adaptive mechanisms, which automatically adjust algorithm parameters during optimization, such as scale factor $F$ and crossover rate $CR$. This adaptability is crucial for archiving efficiency. The proposed CC-SHADE has two main control parameters that need to be set before running. These parameters are the population size and the number of subcomponents.

The main steps of CC-SHADE are as follows:

Step 1: Set the number of subcomponents $M$ and the population size *pop_size*, the maximum number of fitness evaluations ($FEV$),

Step 2: While $FEV > 0$ go to Step 3, otherwise go to Step 6,

Step 3: Set $i$=1 to start a new CC cycle,

Step 4: Evaluate $i$-th subcomponent using SHADE with *pop_size* individuals, $i$++,

Step 5: If $i < M$ then go to Step 3, otherwise go to Step 2,

Step 6: Return the best-found fitness function value and finish the optimization procedure.

In this study, the fitness function is defined as the mean violations $\bar{v}$:

$$\bar{v} = \frac{\sum_{i=1}^{p} G_i(X) + \sum_{j=p+1}^{m} H_j(X)}{m} \quad (4)$$

$$G_i(X) = \begin{cases} g_i(X) \ if \ g_i(X) > 0 \\ 0 \ if \ g_i(X) \leq 0 \end{cases} \quad (5)$$

$$H_j(X) = \begin{cases} |h_j(X)| \ if \ |h_j(X)| - \varepsilon > 0 \\ 0 \ if \ |h_j(X)| - \varepsilon \leq 0 \end{cases} \quad (6)$$

We have extended the constrained real-parameter benchmark [19] up to 1000 variables to evaluate the CC-based approach in solving CSPs in high dimensions, shifted vectors and rotated matrices have been regenerated. We exclude some of the benchmark problems, such as C01, C03, C04, C15, C16, and C20 because any configuration of the CC-SHADE algorithm can find a feasible solution. Thus, we consider 22 benchmark constrained LSGO problems (cLSGO) in this study. The detailed description is presented in Table 1. The first column denotes the names of the cLSGO problem. The second column is divided into two subcolumns, the number of equality and inequality constraints, respectively. Here S means "separable", R means "rotated".

**Table 1.** Detail of benchmark problems in this study. $I$ is the number of inequality constraints, $E$ is the number of equality constraints.

| **Problem** | **The number of constraints** | |
|---|---|---|
| | **E** | **I** |
| C02 | 0 | 1 Non S, R |
| C05 | 0 | 2 Non S, R |
| C06 | 6 | 0 |
| C07 | 2 S | 0 |
| C08 | 2 Non S | 0 |
| C09 | 2 Non S | 0 |
| C10 | 2 Non S | 0 |
| C11 | 1 Non S | 1 Non S |
| C12 | 0 | 2 S |
| C13 | 0 | 3 S |
| C14 | 1 S | 1 S |
| C17 | 1 Non S | 1 S |
| C18 | 1 Non S | 2 Non S |

| | | |
|---|---|---|
| C19 | 0 | 2 Non S |
| C21 | 0 | 2 R |
| C22 | 0 | 3 R |
| C23 | 1 R | 1 R |
| C24 | 1 R | 1 R |
| C25 | 1 R | 1 R |
| C26 | 1 R | 1 R |
| C27 | 1 R | 2 R |
| C28 | 0 | 2 R |

# 4 The results of numerical experiments and discussion

## 4.1 Experimental setup and settings

CC-SHADE and the cLSGO benchmark were implemented in Python. The evaluation of our algorithms was conducted on a computational system running Ubuntu Linux 20.04.6 LTS. A computational cluster, comprising eight PCs equipped with Ryzen 7 2700x (8C/16T) CPUs, was utilized to reduce computation time for numerical experiments. To ensure fair experimental conditions, all benchmark problems were set to a dimension of 1000. The maximum number of fitness evaluations for a single run in each combination (varying *pop_size* and *M*) of CC-SHADE and for each problem was set at 1,000,000 and 25 independent runs. The notation "CC-SHADE(*M/pop_size*)" is used in this study, where *M* represents the total number of subcomponents, and *pop_size* denotes the population size. CC-SHADE was investigated with different population sizes (25, 50, and 100) and varying numbers of subcomponents (1, 2, 4, 8, and 10). This resulted in a total of fifteen combinations for analysis.

## 4.2 Results of numerical experiments

In Figure 1, we can see a heatmap. The X-axis denotes the combination of CC-SHADE, the first number means the number of subcomponents, and the second number means the population size. The Y-axis denotes the number of problems. In each cell, we can see the rank of each CC-SHADE combination based on the best-found median solution on each benchmark problem.

Table 2 displays the median best-found violation values for each problem. The initial column indicates the problem number, the second column presents the performance of non-decomposition approaches, where the number of subcomponents is set to one, and the last column showcases the performance of CC-based approaches. Table 3 provides experimental results using the Wilcoxon test with a significance level set at 0.05. The comparison involves assessing the performance of each CC-SHADE combination against others across all considered constrained large-scale global optimization problems. The first column identifies the CC-SHADE combination, and the second, third, and fourth columns specify how many times the CC-SHADE combination from the corresponding row outperformed, underperformed, or exhibited similar performance compared to other combinations. The fifth column represents the final score for each combination, calculated as the difference between the better and worse values. The last column presents the rank, determined by the final score, with all CC-SHADE combinations sorted accordingly.

Figures 2 and 3 illustrate the convergence trajectories of all CC-SHADE combinations across 11 and 18 benchmark problems, respectively. The X-axis represents the number of fitness evaluations, while the Y-axis represents the averaged best-found fitness values obtained from 25 independent runs.

**Table 2.** Numerical results of the median best-found solutions in 25 independent runs using CC-based and non-decomposition approaches.

| Problem | Non-decomposition approaches | | CC-based approaches | |
|---|---|---|---|---|
| | **The median violations value** | **The population size** | **The median violations value** | **The population size and the number of subcomponents** |
| C02 | 2.61E+06 | 100 | 0.00E+00 | 2/100, 4/(25, 50, 100), 8/(25, 50, 100), 10/(25, 50, 100) |
| C05 | 7.34E+04 | 100 | 0.00E+00 | 4/100, 8(25, 50, 100), 10/(25, 50, 100) |
| C06 | 0.00E+00 | 25,50 | 0.00E+00 | 2/25 |
| C07 | 0.00E+00 | 25 | 0.00E+00 | 2/(25, 50, 100), 4/(25, 50, 100), 8/(25, 50, 100), 10/25 |
| C08 | 5.33E+04 | 100 | 3.54E+04 | 4/100 |
| C09 | 9.54E+00 | 100 | 0.00E+00 | 4/(50, 100), 8(25, 50, 100), 10/(25, 50, 100) |
| C10 | 4.49E+05 | 100 | 7.63E+04 | 8/50 |
| C11 | 3.82E+03 | 100 | 7.99E+02 | 8/100 |
| C12 | 1.30E+03 | 100 | 0.00E+00 | 4/50, 4/100, 8/25, 8/50, 8/100, 10/25, 10/50, 10/100 |
| C13 | 2.60E+04 | 50 | 1.39E+03 | 10/50 |
| C14 | 2.54E+03 | 100 | 0.00E+00 | 4/100, 8/(25, 50, 100), 10/(25, 50, 100) |
| C17 | 5.01E+02 | 100 | 5.01E+02 | 2/(50, 100), 4/(25, 50, 100), 8/(25, 50, 100), 10/(25, 50, 100) |
| C18 | 1.16E+07 | 100 | 6.31E+02 | 10/100 |
| C19 | 7.37499E+05 | 50 | 7.36623E+05 | 8/50 |
| C21 | 1.42E+05 | 100 | 2.86E+04 | 8/100 |
| C22 | 1.91E+05 | 100 | 4.15E+04 | 10/100 |
| C23 | 3.07E+05 | 100 | 5.89E+04 | 8/50 |
| C24 | 9.21E+04 | 100 | 0.00E+00 | 4/(50, 100), 8/(25, 50, 100), 10/(25, 50, 100) |
| C25 | 9.97E+04 | 100 | 0.00E+00 | 4/100, 8/(25, 50, 100), 10/(25, 50, 100) |
| C26 | 1.41E+05 | 100 | 2.65E+04 | 10/50 |
| C27 | 1.44E+10 | 100 | 1.14E+09 | 10/50 |
| C28 | 7.41259E+05 | 100 | 7.41267E+05 | 4/50 |

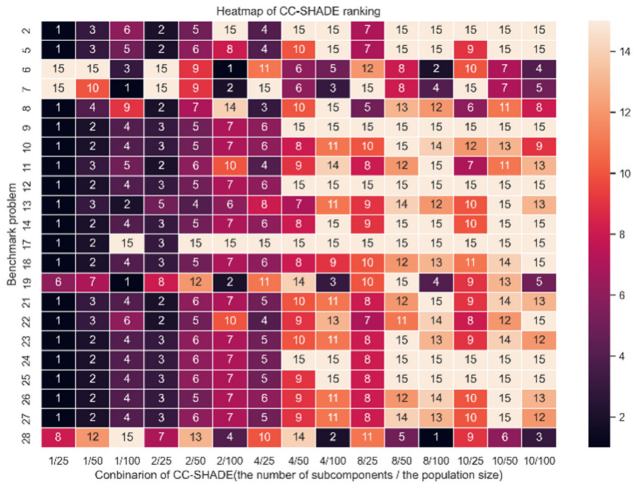

**Fig. 1.** Heatmap based on rankings of different combinations of CC-SHADE.

**Table 3.** Wilcoxon test of different combinations of CC-SHADE.

| Combination | Better (+) | Worse (-) | Equal (≈) | Score | Rank |
|---|---|---|---|---|---|
| 10/50 | 199 | 36 | 95 | 163 | 15 |
| 8/50 | 197 | 40 | 93 | 157 | 14 |
| 8/100 | 182 | 54 | 94 | 128 | 13 |
| 10/100 | 169 | 62 | 99 | 107 | 12 |
| 4/100 | 169 | 84 | 77 | 85 | 11 |
| 10/25 | 166 | 84 | 80 | 82 | 10 |
| 4/50 | 160 | 91 | 79 | 69 | 9 |
| 8/25 | 151 | 108 | 71 | 43 | 8 |
| 2/100 | 120 | 150 | 60 | -30 | 7 |
| 2/50 | 104 | 164 | 62 | -60 | 6 |
| 4/25 | 100 | 165 | 65 | -65 | 5 |
| 1/100 | 69 | 206 | 55 | -137 | 4 |
| 1/50 | 56 | 228 | 46 | -172 | 3 |
| 2/25 | 51 | 232 | 47 | -181 | 2 |
| 1/25 | 22 | 267 | 41 | -245 | 1 |

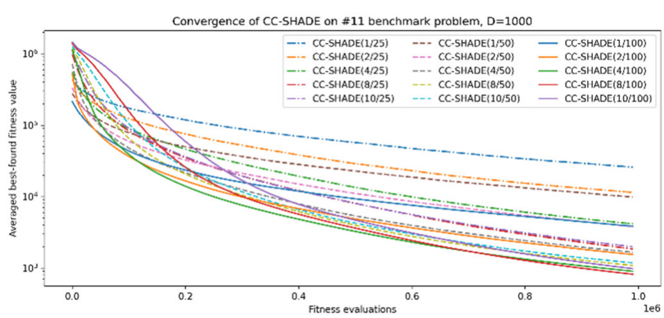

**Fig. 2.** Convergence lines of all considered combinations of CC-SHADE on #11 cLSGO problem.

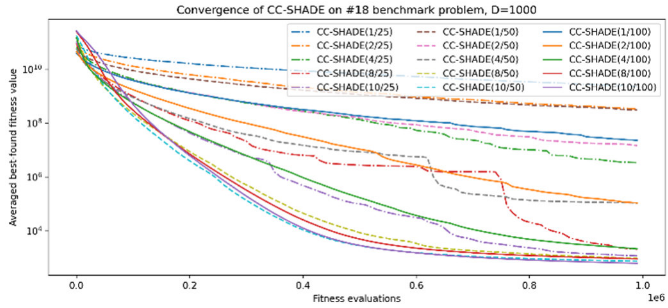


**Fig. 3.** Convergence lines of all considered combinations of CC-SHADE on #18 cLSGO problem.

### 4.3 Discussion of numerical results

As we can see from Table 2, on most benchmark problems, CC-based approaches could find a better solution compared to approaches without decomposition. From Figure 3, we can see that the part of the heating map with a smaller number of subcomponents is mostly darker. It indicates that the combinations with one or a few subcomponents are not able to perform better, on average, than the combinations with more than two subcomponents. Moreover, we are not able to select one of the best combinations of parameters for all benchmark problems. The numerical results from Table 3 prove the previous statement. However, as we can see, the small population size with a big number of subcomponents (8/25) performs worse than the same number of subcomponents but with 50 individuals. Figures 2 and 3 show that different combinations of parameters have different speeds of convergence. It is worth noting that all combinations of CC-SHADE continue to improve the solution, and no stagnation is observed. It is very likely that with an increase in computing resources (more than 1,000,000 of FEVs), it is possible to find better solutions.

## 5 Conclusion

In this study, we assessed the effectiveness of a Cooperative Coevolution-based Evolutionary Algorithm (CC-SHADE) in identifying feasible solution regions within constrained large-scale global optimization problems. The extended benchmark set employed for evaluation featured 1000 variables. The numerical experiments conducted revealed that the performance of CC-SHADE in tackling constrained large-scale global optimization problems is notably influenced by two critical control parameters: the population size and the number of subcomponents. The CC-based evolutionary algorithm demonstrated its capability to identify feasible solution regions in problems characterized by a substantial number of variables. In instances where a feasible region could not be directly located, the algorithm showcased the ability to minimize constraint violations, progressively approaching the feasible region. The proposed benchmark set serves as a valuable tool for evaluating the performance of other meta-heuristic algorithms designed for constraint handling. In future investigations, our focus will shift towards developing an algorithm capable of concurrently addressing both constraints and objective functions. This endeavor is motivated by the recognition that constraints and objectives often necessitate distinct decomposition strategies.

This work was supported by the Ministry of Science and Higher Education of the Russian Federation (Grant No.075-15-2022-1121).

## References

1. F. Vaz, Y. Lavinas, C. Aranha, M. Ladeira, *Exploring constraint handling techniques in real-world problems on MOEA/D with limited budget of evaluations*, in Proceedings of Evolutionary Multi-Criterion Optimization: 11th International Conference, EMO, March 28–31 2021, Shenzhen, China (2021)
2. S.C. Brailsford, C.N. Potts, B.M. Smith, European journal of operational research **119(3)**, 557-581 (1999)
3. M.A. Potter, K.A.D. Jong, Evolutionary computation **8**, 1-29 (2000)
4. A. Neumaier, Acta numerica **13**, 271-369 (2004)
5. A.E. Eiben, *Evolutionary algorithms and constraint satisfaction: Definitions, survey, methodology, and research directions*, Theoretical aspects of evolutionary computing, 13-30, 2001.
6. M. Ionita, M. Breaban, C. Croitoru, New Achievements in Evolutionary Computation **17**, (2010)
7. N. Sakamoto, Y. Akimoto, *Adaptive ranking based constraint handling for explicitly constrained black-box optimization*, in Proceedings of the Genetic and Evolutionary Computation Conference, 700-708 (2019)
8. H. Kanoh, M. Matsumoto, K. Hasegawa, N. Kato, S. Nishihara, Engineering Applications of Artificial Intelligence **10(6)**, 531-537 (1997)
9. J. Liu, W. Zhong, L. Jiao, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **36(1)**, 54-73 (2006)
10. O. Kramer, Applied Computational Intelligence and Soft Computing **2010**, 1-19, (2010)
11. V.H. Cantú, C. Azzaro-Pantel, A. Ponsich, Applied Soft Computing **108**, 107442, (2021)
12. J. Lampinen, *A constraint handling approach for the differential evolution algorithm*, in Proceedings of the 2002 Congress on Evolutionary Computation, CEC'02 (Cat. No. 02TH8600) **2** (2002)
13. M. Breaban, M. Ionita, C. Croitoru, *A new PSO approach to constraint satisfaction*, in 2007 IEEE Congress on Evolutionary Computation, 1948-1954, (2007)
14. I. Lin, *Particle swarm optimization for solving constraint satisfaction problems*, Master's thesis, Simon Fraser Univ. (2005)
15. B.G. Craenen, A.E. Eiben, IEEE Congress on Evolutionary Computation **3**, 1922-1928, (2005)
16. X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, Z. Zhu, IEEE Transactions on Evolutionary Computation **23(3)**, 421-441 (2018)
17. A. Vakhnin, E. Sopov, Algorithms **14(146)** (2021)
18. R. Tanabe, A. Fukunaga, *Success-history based parameter adaptation for differential evolution*, In 2013 IEEE congress on evolutionary computation, 71-78, (2013)
19. G. Wu, R. Mallipeddi, P.N. Suganthan, *Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization*, Technical Report (2017)