F O U N D A T I O N S O F C O M P U T I N G A N D D E C I S I O N S C I E N C E S
Vol. 44 (2019) No. 2
ISSN 0867-6356
DOI: 10.2478/fcds-2019-0008 e-ISSN 2300-3405

sciendo

Benchmarking of Problems and Solvers: a Game-Theoretic Approach

Joseph Gogodze*

Abstract. In this note, we propose a game-theoretic approach for benchmarking computational problems and their solvers. The approach takes an assessment matrix as a payoff matrix for some zero-sum matrix game in which the first player chooses a problem and the second player chooses a solver. The solution in mixed strategies of this game is used to construct a notionally objective ranking of the problems and solvers under consideration. The proposed approach is illustrated in terms of an example to demonstrate its viability and its suitability for applications.

Keywords: benchmarking, software, solvers, problems, testing, multi objective decision-making problem

1. Introduction

In recent years, evaluating the performance of solvers has become a topic of intense study, and various approaches have been discussed in the literature. Most benchmarking tests produce tables showing each solver's performance for each problem according to a specified evaluation metric, such as the central processing unit (CPU) time, number of function evaluations, or number of iterations. Interpretations of this data, i.e., the selection of benchmarking method, currently depends on the subjective tastes and preferences of individual researchers, who evaluate solvers using different problem sets and metrics, [2-5,7,8,10,13,14,20].

* Institute of Control System, TECHINFORMI, Georgian Technical University, 77 Kostava str., , 0175 Tbilisi, Georgia. Email: sosogogodze@gtu.ge

This study introduces a new benchmarking approach that explores the natural relations between problems and solvers as determined by their evaluation tables. Specifically, we present data for benchmarking in the form of a benchmarking context, i.e., as a triple $< S, P, J >$ where S and P are sets of solvers and problems, respectively and $J: S \times P \to \mathbb{R}$ is an assessment function (a performance metric or evaluation metric). This concept is quite general and emphasizes that problem and solver benchmarking cannot be considered independently. Throughout the paper the benchmarking context $< S, P, J >$ assumes that the sets of solvers and problems are finite.

The benchmarking procedure proposed in this study uses the data encapsulated by the given benchmarking context$< S, P, J >$ and a new multi-objective decision making (MODM) procedure. Specifically, we consider the set of problems as a set of alternatives and the set of solvers as a set of criteria. Now, we can define a decision matrix as a matrix whose elements exhibit the performance of different alternatives (i.e., solvers) with respect to various criteria (i.e., problems) through the assessment function.

The rationale of such consideration is that such a multi-objective formulation allows us to use the concept of Pareto optimality and a vast arsenal of different approaches for Pareto optimization. However, it must be stressed that a characteristic feature of Pareto optimality is that the set of Pareto-optimal alternatives is large and that all Pareto-optimal alternatives must be considered as mathematically equal. On the other hand, because decisions are usually expected to be unique, additional factors are considered for selecting specific and/or in some sense more appropriate alternatives from the set of Pareto-optimal alternatives. Our approach allows us to select, in some sense, appropriate solvers. The essence of the method lies in the fact that an (in some sense) objective weighting method for the MODM problem described above can be obtained solving a special two-person zero-sum game. Specifically, we can consider the decision matrix described above as a pay-off matrix for some zero-sum matrix game in which the S-player chooses one of the solvers (i.e., alternatives) from S and the P-player chooses one of the problems (i.e., criteria) from P. Solving this game using mixed strategies, we can find both appropriate solvers and reasonable rankings of solvers and problems at the same time.

We demonstrate the possibilities of the proposed method on a concrete example. Specifically, as an illustrative example, based on data from [16], we benchmark 9 differential evolutionary algorithms on a set of 50 test problems, in accordance with the $\mathrm{ERT}_{\mathrm{RSE}}$ (Random Sampling Equivalent Expected Run Time) performance metric.

The remainder of this paper is organized as follows. Section 2 describes the proposed methodology for evaluating and comparing solvers and problems. Section 3 considers the applications of the proposed tool in a selected benchmarking problem. Section 4 draws conclusions.

2. Proposed Methods

Throughout the paper, $\mathbb{R}^n$ is $n-$dimensional space, with norm $||\cdot||$ and scalar product $<\cdot,\cdot>$. Moreover, we use the following notations for the special sets and special vectors:

$$\mathbb{R}_+^n = \{\xi \in \mathbb{R}^n | \xi_k \geq 0, k = 1, \dots, n\},\ \ \Delta_n = \{\xi \in \mathbb{R}_+^n | \sum_{k=1}^n \xi_k = 1\},$$
$$e_k = \left(0, \dots, 1_{(k)}, \dots, 0\right) \in \mathbb{R}^n, k = 1, \dots, n;\ 1_n = (1, \dots, 1) \in \mathbb{R}^n;\ 0_n = (0, \dots, 0) \in \mathbb{R}^n.$$

2.1. Preliminaries regarding MODM problems

We begin here with notations and definitions that are necessary for our further considerations. We assume that the initial data for decision-making are presented as a decision matrix, X, the elements of which exhibit the performance of diverse alternatives with respect to various criteria:

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{bmatrix}$$

where x_{ij} the performance is measure of the i-th alternative on the j-th criterion $(i = 1, .. n, j = 1, \dots, n)$, m is the number of alternatives, and n is the number of criteria.

For simplicity, we assume further that each criterion can be treated as non-beneficial (for which lower values are preferable) criteria. We assume in addition that the decision matrix is normalized for ensuring the comparability of its elements and is such that $\sum_{k=1}^n x_{kj} > 0$, $x_j^{min} = \min\{x_{1j}, \dots, x_{mj}\} = 0$, $x_j^{max} = \max\{x_{1j}, \dots, x_{mj}\} = 1$ for all $j = 1, ...,$n i.e. elements of the matrix X are dimensionless numbers representing the normalized performance of the i-th alternative on the j-th criterion $(i = 1, .. n, j = 1, \dots, n)$. Thus, the decision-making procedure's goal after the normalization procedure is simultaneous minimization of all criteria, i.e., we obtain a typical multi-objective optimization problem.

Recall now the basic concepts of multi-objective optimization theory. To this end we introduce the following notations: alternatives will be denote by $a_i, i = 1, \dots, m,$ $A = \{a_1, \dots, a_m\}$ and criteria by $c_j: A \longrightarrow \mathbb{R}, j = 1, \dots, n$ with the result that $x_{ij} = c_j(a_i)$, $i = 1, .. m, j = 1, \dots, n$. Further, the set A will be designated as the set of alternatives, map $\vec{c} = (c_1, \dots, c_n)$ will be designated as the criterion map (correspondingly c_j is the j-th objective $j = 1, ...,$n), and the set $\vec{c}(A)$ will be designated as the set of admissible criterion values. The following concepts are also associated with the criterion map and the set of alternatives. We say that an alternative $\bar{a} \in A$ is a minimizer of the j-th criterion, if $c_j(\bar{a}) = \min_{a \in A} c_j(a)$. We denote by $A_{min}^j(\vec{c})$ the set of all minimizers of the j-th objective, $j = 1, \dots,$ n. Correspondingly, a point$\xi_j = \vec{c}(a) \in \vec{c}(A)$, where $a \in A_{min}^j(\vec{c})$ will be designated as an anchor point, $j = 1, ...,$n. The point $\xi^I = (\xi_1^I, \dots, \xi_n^I)$ where if $\xi_j^I = \min_{a \in A} c_j(a)$, objective $j = 1, ...,$n, will be designated as an ideal point. We say that an ideal point is attainable if there exists an alternative $a^I \in A$ such that $\xi^I = \vec{c}(a^I)$.

We say that point $a_* \in A$ is Pareto-optimal (efficient) if there is no $a \in A$ with $c_j(a) \leq c_j(a_*)$ for all $j = 1, ...,$n, and there exists an index $j_0 \in \{1, \dots, n\}$ such that $c_{j_0}(a) < c_{j_0}(a_*)$. The set of all efficient alternatives will be denoted A_e and designated as a Pareto set Accordingly, $\vec{c}(A_e)$ will be designated as an efficient front.

A very common method for solving the MODM problem is the weighting method. This method requires specifying how we are to determine what we consider to be objective weights. To this end, for simplicity we consider here only entropy method (ENT), [22]. To determine objective weights using the entropy measure, the decision matrix must to be normalized as follows:

$$p_{ij} = x_{ij} / \sum_{k=1}^{m} x_{kj} \; (i = 1, .. n, j = 1, \dots, n).$$

Now, the amount of decision information contained in the matrix p_{ij} and emitted from each criterion can be measured by the entropy values

$$e_j = -\frac{1}{lnm} \sum_{k=1}^{m} p_{kj} \ln (p_{kj}), \qquad j = 1, \dots, n.$$

Correspondingly, the degree of divergence $d_j = 1 - e_j$ can be defined and the objective weight can be calculated as

$$w_j = d_j \Big/ \sum_{k=1}^{m} d_k, \;\; j = 1, \dots, n.$$

It is possible to determine objective weights similarly for alternatives, but we omit the corresponding details here.

2.2. Game-theoretic approach for solving MODM problems

Here, we present our game-theoretic approach for solving MODM problems, [11]. The proposed method considers the matrix X as a pay-off matrix for some zero-sum matrix game. We can interpret this game as follows. The row-player (A-player) chooses one of the alternatives $a \in A$, and the column-player (C-player) chooses one of the criteria $c \in C$. The quantity $x_{ij} = c_j(a_i)$ is a sum paid to the A-player by the C-player when the A-player chooses the alternative $a_i \in A$, and the C-player chooses the criteria $c_j \in C$, $i = 1, .. m, j = 1, \dots, n$. The mixed strategy for the A-player is a vector $\xi \in \Delta_m$, and the mixed strategy for the C-player is a vector $\zeta \in \Delta_n$. Correspondingly, components $\xi_k, k = 1, \dots, m$, ($\zeta_k, k = 1, \dots, n$) are the probabilities of choices for alternative k by the A-player (resp., criteria k by the C-player). Hence, for mixed strategies $\xi \in \Delta_m$, $\zeta \in \Delta_n$, the expected pay-off for the A-player is

$$\Lambda(\xi, \zeta) = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} \xi_i \zeta_j = \sum_{i=1}^{m} P_{\zeta}(a_i) \xi_i = \sum_{j=1}^{n} Q_{\xi}(c_j) \zeta_i \,,$$

where we use the notation

$$P_{\zeta}(a) = \sum_{j=1}^{n} c_j(a), \forall a \in A; \;\; Q_{\xi}(c) = \sum_{i=1}^{m} c(a_i) \;\; \forall c \in C.$$

Note that $P_\zeta(a)$ can be interpreted as the expected pay-off of the alternative $a \in A$, when choosing the C-player's mixed strategies $\zeta \in \Delta_n$ and $Q_\xi(c)$ can be interpreted as expected pay-offs for the criterion $c \in C$ when choosing the A-player's mixed strategies $\xi \in \Delta_m$. Recall also that a pair of mixed strategies $(\xi^*, \zeta^*) \in \Delta_m \times \Delta_n$ is a Nash equilibrium solution of the considered zero-sum matrix game if and only if

$$\max_{\xi \in \Delta_m} \min_{\zeta \in \Delta_n} \Lambda(\xi, \zeta) = \min_{\zeta \in \Delta_n} \max_{\xi \in \Delta_m} \Lambda(\xi, \zeta) = \Lambda(\xi^*, \zeta^*).$$

Let $(\xi^*, \zeta^*) \in \Delta_m \times \Delta_n$ be a solution of the considered zero-sum matrix game. Now, we interpret $\zeta^* \in \Delta_n$ as a properly chosen weight and consider $P^{\vec{c}}_{\xi^*}(a) = \sum_{j=1}^n c_j(a)\,\zeta_j^*$ as a true aggregation of performance criteria. Moreover, it is well-known that any solution of this minimization problem

$$\left.\begin{array}{c} P^{\vec{c}}_{\xi^*}(a) = \sum_{j=1}^{n} c_j(a)\,\zeta_j^* \longrightarrow min \\ a \in A \end{array}\right\}$$

is always Pareto optimal, and hence, the presented approach allows the selection of some Pareto-optimal alternative, which can be considered as appropriate.

The relevant interpretation of the procedure described above is necessary for determining in what sense this obtained Pareto-optimal alternative is appropriate. We assume that the A -player and C-player are represented by the populations designated as the A -population and C-population, respectively. Moreover, we also assume that to each alternative (criterion) there corresponds the subpopulation of individuals that dispose of this and only this alternative (criterion) in the A-population (resp., C-population), and that such subpopulations cover the entire A-population (resp., C-population). We also interpret component $\xi_k, k = 1, \dots, m,$ $(\zeta_k, k = 1, \dots, n)$ of a mixed strategy $\xi \in \Delta_m (\zeta \in \Delta_n)$ as sharing in a corresponding subpopulation in the A-population (resp., C-population).

It is useful to be able to compare the proposed method ,with some known method of MODM-problem solution. For this purpose, we use the ENT method described in the previous subsection. After the weights determining by the ENT method or by the GTR method, the criteria can be aggregated/scalarized into the single criterion. The aggregate criterion's value on an alternative defines rating of this alternative. The subsequent ordering (by ascending or descending ratings-at wish) of the alternatives determines their ENT-ranking or GTR-ranking, respectively.

2.3. Benchmarking problem

Consider a set P of problems, a set S of solvers, and a function $J: S \times P \to \mathbb{R}$ - the assessment function (performance metric). Further, assume for definiteness that the high and low values of J correspond to the worst and best cases, respectively, and for convenience, interpret $J(s,p)$ as the cost of solving the problem $p \in P$ with solver problem $s \in S$. Note that if $J(s,p) < J(s',p')$, then it can be said that $s \in S$ solves problem $p \in P$ better than solver $s' \in S$ solves problem $p' \in P$ (i.e., the problem $p \in P$ was easier for solver $s \in S$ than the problem $p' \in P$ was for solver $s' \in S$).

For a given benchmarking context $< S, P, J >$ assume further that the following assumptio,ns hold (where n_P, n_S are given natural numbers):

$$\begin{cases} (A0) & P = \{1, \dots, n_P\}, S = \{1, \dots, n_S\}; \\ (A1) & J(s,p) \geq 0 \; \forall (s,p) \in S \times P \\ (A2) & I_P = \sum_{p \in P} J(s,p) > 0 \quad \forall s \in S \end{cases}$$

Assumption (A0) establishes that the sets S, P are finite. Assumption (A2) can be interpreted as a no-triviality condition of the assessments (such that, as a requirement, each solver and each problem are to be tested with at least one problem and one solver, respectively). (A2) implies that $I_{SP} = \sum_{s \in S, p \in P} J(s,p) > 0$. The triple $< S, P, J >$, which satisfies assumptions (A0), (A1), and (A2), is henceforth referred to as the benchmarking context.

Let us assume now that $< S, P, J >$ is a given benchmarking context. Define the set of alternatives as S i.e., $A = S = \{s_1, \dots, s_{n_S}\}$. At the same time, define the criteria set $C = \{c_1, \dots, c_{n_P}\}$ as follows: $c_j(\cdot) = J(\cdot, p_j): S \to \mathbb{R}$ for $p_j \in P, j = 1, \dots, n_P$; i.e., we can assume that $C = P$. Hence, we obtain decision matrix $X = [x_{ij}]$, where $x_{ij} = c_j(a_i) = J(s_i, p_j)$, $i = 1, .. n_S, j = 1, \dots, n_P$. We assume also that $x_j^{min} < x_j^{max}$ for all $j = 1, \dots, n_P$. Now we can apply the game-theoretic approach for solving MODM problems, as it was described in the previous subsection.

3. Benchmarking of Differential Evolution Algorithms

We focus in this section on an illustrative example using the proposed method. To apply the proposed approach, we must solve the corresponding zero-sum game. To this end, we use the standard approach of reducing this game-theoretic problem to a linear programming problem. All calculations were performed using the MATLAB environment.

Note finally that, as we will see below, the results are quite appropriate and competitive and were found without any previous estimates of the importance of the criteria. For comparison, we also present here results obtained using the ENT method described above (see subsection 2.1).

The study [16] considered the 9 optimization algorithms and 25 test functions and correspondingly we consider the sets of 9 solvers and 50 problems (see Tables B.1– B4, Annex B.) The sources cited in these tables present detailed information on the selected algorithms and test functions. The $\mathrm{ERT}_{\mathrm{RSE}}$ metric/assessment function used in [16] (see Annex A for detail description).

Table B5, Annex B, presents the $\mathrm{ERT}_{\mathrm{RSE}}$ values for all problem-solver pairs. Obviously, assumptions (A0), (A1), and (A2) hold in the case under consideration, and hence, the benchmarking context $< S, P, J >$ is fully determined.

Here we present the ranking results of the solvers/problems obtained using the proposed method, GTR, and, for comparison, the results obtained using the ENT method. First, note

that the solution (equilibrium) in the mixed strategies for the corresponding zero-sum game is

$$\xi^* = (0.2299, 0.2636, 0, 0, 0, 0.1976, 0, 0.3089, 0)$$

$$\zeta^* = (0, 0, 0.2037, 0, 0, 0, 0.3000, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,$$
$$0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.2960, 0, 0, 0, 0.2003, 0, 0, 0, 0, 0, 0)$$

This solution can be interpreted as the distribution of some population according to preferences regarding solvers/problems (see subsection 2.2). Note also that using the proposed method, within a given benchmarking context, the solvers S01, S02, S06, S08 and the problems P03, P07, P40, P44 have a special status (are significant).

The results of the ranking of solvers and problems using the three methods indicated above are reported in Tables 1 and 2 and illustrated in Figures 1 and 2, respectively. Our calculations show that according to the GTR-ranking the solvers S01 and S09 (the problems P07 and P20) can be considered as “the first” and “the last”, respectively, in the framework of the considered benchmarking context.

Table 1. Ranking solvers using different methods

Solver	ENT	GTR
S01	8	9
S02	9	6
S03	5	5
S04	2	3
S05	4	4
S06	6	7
S07	7	2
S08	3	8
S09	1	1

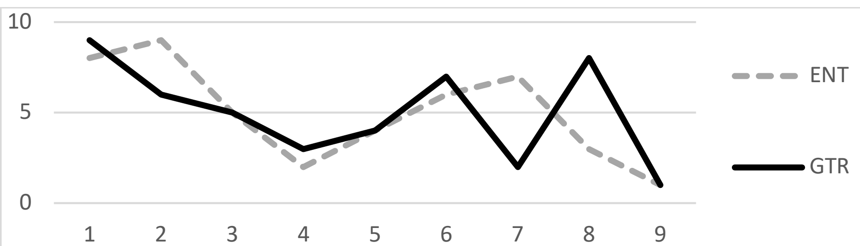

Figure 1. Comparison of rankings obtained using different methods.

Vertical axis-Rank, Horizontal axis–Solvers (see main text for explanation)

Table 2. Ranking problems using different methods

Problem	ENT	GTR		Problem	ENT	GTR
P01	10	13		P26	6	9
P02	34	27		P27	33	34
P03	4	2		P28	26	10
P04	24	14		P29	29	16
P05	38	37		P30	21	15
P06	22	24		P31	19	33
P07	1	1		P32	2	5
P08	49	48		P33	50	39
P09	12	23		P34	14	32
P10	8	12		P35	9	36
P11	48	35		P36	47	40
P12	45	45		P37	44	47
P13	16	20		P38	30	31
P14	39	30		P39	36	29
P15	46	42		P40	40	3
P16	17	7		P41	25	28
P17	37	46		P42	23	18
P18	43	49		P43	15	17
P19	42	41		P44	5	4
P20	41	50		P45	13	25
P21	7	11		P46	20	22
P22	27	21		P47	28	26
P23	11	19		P48	31	38
P24	35	43		P49	32	44
P25	18	8		P50	3	6

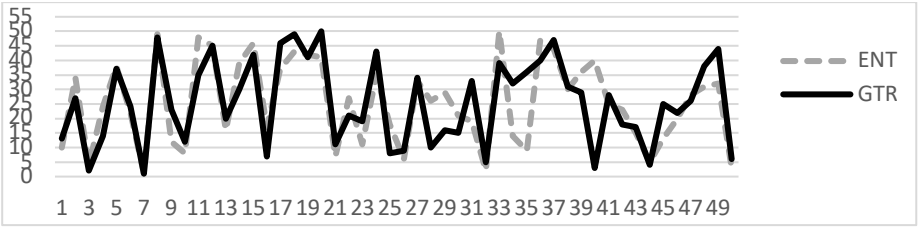

Figure 2. Comparison of rankings obtained using different methods

Vertical axis-Rank, Horizontal axis–Problems (see main text for explanation)

4. Conclusion

In this paper, we proposed a new method for benchmarking computational problems and their solvers. The method is based on a game-theoretic approach to the solving of MODM problems. The proposed method is sufficiently general to be used in other areas. Furthermore, we considered an illustrative example to demonstrate the viability and suitability of the proposed method for applications.

Note also that we only provided a tool for benchmarking a given set of solvers (problems) on a given set of problems (solvers) using a given performance metric, i.e., in the framework of the given benchmarking context. However, issues regarding selection of benchmarking context components remain open. The literature does not contain clear and direct recommendations regarding how to select a collection of solvers, problems, and performance metrics to obtain benchmarking results with proper justification. Hence, further development of benchmarking methods must focus to a greater extent on the justification of a benchmarking context's component choices.

Annex A

A description of the assessment function used in [16] follows. First, note that the expected running time (ERT), a widely used performance metric for optimization algorithms, is defined as

$$ERT(\tau) = mean(M_\tau) + \frac{1-q}{q} N_{max}, q = \frac{N_{succes}}{N_{total}}$$

where τ is a reference threshold value, M_τ is the number of function evaluations required to reach an objective value better than τ (such as successful runs), N_{max} is the maximum number of function evaluations per optimization run,N_{succes} is the number of successful runs, N_{total} is the total number of runs, and q is the named success rate [1].

Note that ERT can be interpreted as the expected number of function evaluations of an algorithm to reach an objective function threshold for the first time and for the ERT performance measure, a threshold or success criterion is required. However, unlike conventional optimization problems (where ERT criterion is usually related to reaching the value of the known global optimum, within a specified tolerance), for difficult optimization problems the probability of coming close to the global optimum is negligible and a more acceptable alternative success criterion is required. Moreover, to compare qualitative performance using ERT for the difficult optimization problems, it is necessary that all compared algorithms meet the success criterion at least a few times. Correspondingly, in [16] was used the success criterion as reaching a target value which corresponds to the expected value of the best objective function value obtained from uniform random sampling (1000 samples). Next, for a test function f determined the expected objective value $E^{RSE}(f)$, the estimation of which is based on the 100 repetitions. Finally, the ERT w.r.t. this objective function value limit was referred as ERT_{RSE} (Random Sampling Equivalent-Expected Run Time) for the test function f.

Annex B

Table B.1. Solvers

Solver	S01	S02	S03	S04	S05	S06	S07	S08	S09
Algorithm	DE	DE2	jDE	JADE	SADE	Code	epsDE	SQG	SQG-DE

Table B.2. Problems

Problem	Description		Problem	Description	
	Dimension	Function		Dimension	Function
P01	30	F01	**P26**	50	F01
P02	30	F02	**P27**	50	F02
P03	30	F03	**P28**	50	F03
P04	30	F04	**P29**	50	F04
P05	30	F05	**P30**	50	F05
P06	30	F06	**P31**	50	F06
P07	30	F07	**P32**	50	F07
P08	30	F08	**P33**	50	F08
P09	30	F09	**P34**	50	F09
P10	30	F10	**P35**	50	F10
P11	30	F11	**P36**	50	F11
P12	30	F12	**P37**	50	F12
P13	30	F13	**P38**	50	F13
P14	30	F14	**P39**	50	F14
P15	30	F15	**P40**	50	F15
P16	30	F16	**P41**	50	F16
P17	30	F17	**P42**	50	F17
P18	30	F18	**P43**	50	F18
P19	30	F19	**P44**	50	F19
P20	30	F20	**P45**	50	F20
P21	30	F21	**P46**	50	F21
P22	30	F22	**P47**	50	F22
P23	30	F23	**P48**	50	F23
P24	30	F24	**P49**	50	F24
P25	30	F25	**P50**	50	F25

Table B.3. Test Functions

Function	Short Description (see [23])
F01	Shifted Sphere Function
F02	Shifted Schwefel's Problem 1.2
F03	Shifted Rotated High Conditioned Elliptic Function
F04	Shifted Schwefel's Problem 1.2 with noise in fitness function
F05	Schwefel's Problem 2.6 with the global optimum on the bounds
F06	Shifted Rosenbrock's Function
F07	Shifted Rotated Griewank's Function
F08	Shifted Rotated Ackley's Function with the global optimum on the bounds
F09	Shifted Rastrigin's Function
F10	Shifted Rotated Rastrigin's Function
F11	Shifted Rotated Weierstrass Function
F12	Schwefel's Problem 2.13
F13	Expanded Extended Griewank's plus Rosenbrock's Function
F14	Shifted Rotated Expanded Scaffer's F6
F15	Hybrid Composition Function
F16	Rotated Hybrid Composition Function
F17	Rotated Hybrid Composition Function
F18	Rotated Hybrid Composition Function
F19	Rotated Hybrid Composition Functions with noise in fitness function
F20	Rotated Hybrid Composition Function with a narrow basin for the global optimum
F21	Rotated Hybrid Composition Function
F22	Rotated Hybrid Composition Function with a high condition number matrix
F23	Non-Continuous Rotated Hybrid Composition Function
F24	Rotated Hybrid Composition Function
F25	Rotated Hybrid Composition Function

Source: [16].

Table B.4. Algorithms

Algorithm	Short Description	Source
DE	"Rand/1/bin" Differential Evolution	[17]
DE2	"Best/2/bin" Differential Evolution	[18]
jDE	Self-adapting Differential Evolution	[6]
JADE	Adaptive Differential Evolution	[23]
SaDE	Strategy adaptation Differential Evolution	[15]
Code	Composite trial vector strategy Differential Evolution	[21]
epsDE	Ensemble parameters Differential Evolution	[12]
SQG	Stochastic Quasi-Gradient search	[9]
SQG-DE	Stochastic Quasi-Gradient based Differential Evolution	[16]

Source: [16].

Table B5. ERT_{RSE} Metric

Problems	Solvers								
	S01	**S02**	**S03**	**S04**	**S05**	**S06**	**S077**	**S08**	**S09**
P01	11657	24049	491	264	329	904	606	202	168
P02	7401	11567	739	309	379	894	3653	137	318
P03	6536	32763	606	356	519	1148	1045	135	210
P04	6182	13326	570	285	342	662	1712	694	310
P05	4342	4766	536	311	527	871	564	459	160
P06	7404	11728	491	234	315	807	603	124	168
P07	256	124	482	310	596	879	169	32365	107
P08	2877	3011	1783	2084	2673	2720	2082	132	2771
P09	15735	24058	467	269	380	818	726	126	193
P10	11658	24054	474	256	320	850	630	201	172
P11	2414	1555	1967	1502	1321	1541	1979	368	1949
P12	1072	934	509	342	416	931	594	128	225
P13	15709	10150	524	224	231	723	898	161	289
P14	7497	19174	2735	1735	1509	2366	6379	6899	1658
P15	437	373	668	366	565	857	458	1016	185
P16	2999	11671	490	335	615	689	616	219	179
P17	6925	11670	514	362	572	845	693	9096	178
P18	1017	1036	501	314	460	741	341	263	143
P19	1271	1045	533	317	551	836	389	259	148
P20	1190	1098	491	315	545	1039	384	284	154
P21	9466	24251	498	253	327	901	597	301	175
P22	2433	2889	489	281	389	730	602	13320	202
P23	13650	24194	474	254	325	802	586	664	181
P24	7785	4449	544	272	376	911	573	10183	187
P25	220	115	445	304	554	855	167	5321	109
P26	19030	9054	439	205	254	674	579	192	178
P27	9055	8145	572	316	388	908	2135	127	296
P28	3819	10317	514	289	455	764	566	121	145
P29	19059	10175	538	341	434	920	2931	1332	310
P30	9173	19040	568	294	366	964	1418	451	214
P31	11557	13327	491	235	244	831	609	132	182
P32	394	187	443	289	536	884	231	24041	109
P33	1732	2347	2098	1792	2724	2522	2740	117	1960
P34	15719	19060	482	245	295	845	738	123	194
P35	24071	24059	456	235	306	739	661	169	188
P36	2024	2568	2187	1324	1814	1825	1675	347	1331
P37	1183	1263	560	340	418	1132	774	119	224
P38	7381	9057	514	227	193	604	1139	145	336
P39	7476	15824	1792	1310	1198	1296	3482	4706	1808
P40	341	291	557	354	556	852	331	205	173
P41	7506	11568	495	320	487	738	593	260	156
P42	11585	32369	640	327	515	794	809	5743	192
P43	10264	19135	631	319	380	1071	892	352	226
P44	24351	7387	511	290	389	866	728	307	210
P45	19307	13407	533	321	372	915	794	380	227

(Table B5, continued)

Problems	Solvers								
	S01	**S02**	**S03**	**S04**	**S05**	**S06**	**S077**	**S08**	**S09**
P46	13571	9089	486	243	320	871	605	325	176
P47	3311	5068	564	291	431	755	724	19048	153
P48	7468	7544	483	248	333	880	590	537	170
P49	15709	13335	498	231	245	784	688	8170	201
P50	304	163	488	315	552	1038	210	13368	109

Source: [16].

References

[1] Auger, A., Hansen, N., Performance evaluation of an advanced local search evolutionary algorithm, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2, 2005, 1777-1784.

[2] Benson, H.Y., Shanno, D.F., Vanderbei, R.J., Interior-point methods for nonconvex nonlinear programming: Jamming and comparative numerical testing, Operations Research and Financial Engineering, Princeton University, Technical Report ORFE-00-02, 2000.

[3] Billups, S.C., Dirkse, S.P., Ferris, M.C., A comparison of algorithms for large-scale mixed complementarity problems, Comput. Optim. Appl., 7,1997, 3–25.

[4] Bondarenko, A.S., Bortz, D.M., More, J.J., COPS: Large-scale nonlinearly constrained optimization problems, No. ANL/MCS-TM-237. Argonne National Lab., IL (US), 2000.

[5] Bongartz, I., Conn, A.R., Gould, N.I.M., Saunders, M.A., Toint, P.L., A numerical comparison between the LANCELOT and MINOS packages for large-scale numerical optimization, Report 97/13, Namur University, 1997

[6] Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V., Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, IEEE transactions on evolutionary computation, 10, 6, 2006, 646-657.

[7] Conn, A.R., Gould, N.I.M., Toint, P.L, Numerical experiments with the LANCELOT package (Release A) for large-scale nonlinear optimization, Math. Program., 73, 1996, 73–110.

[8] Dolan, E.D., Moré, J.J., Benchmarking optimization software with performance profiles. Mathematical programming 91.2,2002, pp. 201-213.

[9] Ermoliev, Y. M., Methods of solution of nonlinear extremal problems, Cybernetics, 2,4, 1966, 1-14.

[10] Gogodze J., PageRank method for benchmarking computational Problems and their solvers, International Journal of Computer Science Issues, 15, 3, 2018, 1-7.

[11] Gogodze J., Using a Two-Person Zero-Sum Game to Solve a Decision-Making Problem, Pure and Applied Mathematics Journal, 7, 2, 2018, 11-19.

[12] Mallipeddi, R., Suganthan, P. N., Pan, Q. K., Tasgetiren, M. F., Differential evolution algorithm with ensemble of parameters and mutation strategies, Applied Soft Computing, 11, 2, 2011, 1679-1696.
[13] Mittelmann, H., Benchmarking interior point LP/QP solvers, Optim. Methods Softw., 12, 1999, 655–670.
[14] Nash, S.G., Nocedal, J., A numerical study of the limited memory BFGS method and the truncated Newton method for large scale optimization, SIAM J. Optim., 1, 1991, 358–372
[15] Qin, A. K., Huang, V. L., Suganthan, P. N., Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE transactions on Evolutionary Computation, 13,2, 2009, 398-417.
[16] Sala, R., Baldanzini N., Pierini M., SQG-Differential Evolution for difficult optimization problems under a tight function evaluation budget, in: International Workshop on Machine Learning, Optimization, and Big Data, Springer, Cham, 2017, 322-336.
[17] Storn, R., Price, K., Differential Evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, Berkeley: ICSI ,1995.
[18] Storn, R., Price, K., Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. of global optimization, 11,4, 1997, 341-359.
[19] Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S., Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, KanGAL report, 2005005, 2005.
[20] Vanderbei, R.J., Shanno, D.F., An interior-point algorithm for nonconvex nonlinear programming, Comput. Optim. Appl., 13, 1999, 231–252.
[21] Wang, Y., Cai, Z., Zhang, Q., Differential evolution with composite trial vector generation strategies and control parameters, IEEE Transactions on Evolutionary Computation, 15,1, 2011, 55-66.
[22] Zeleny M., Multiple criteria decision making, New York: McGraw-Hill, 1982.
[23] Zhang, J., Sanderson, A. C., JADE: adaptive differential evolution with optional external archive, IEEE Transactions on evolutionary computation, 13,5, 2009, 945-958.

Received 18.07.2018, Accepted 14.01.2019