International Journal of Mathematics and
Computer Science, **19**(2024), no. 4, 987–997

Adaptive differential evolution with feasibility ranking mutation and dynamic thresholding technique for constrained optimization

Wittaya Phaengthaisong, Jeerayut Wetweerapong, Pikul Puphasuk

Department of Mathematics
Faculty of Science
Khon Kaen University
Khon Kaen 40002, Thailand

email: wittaya_paengthaisong@kkumail.com, wjeera@kku.ac.th, ppikul@kku.ac.th

(Received February 1, 2024, Accepted March 18, 2024, Published June 1, 2024)

Abstract

Real-world optimization problems in engineering and applied sciences often involve constraints and require effective methods using constraint-handling techniques to solve them. We propose an adaptive differential evolution algorithm with feasibility ranking mutation and dynamic thresholding technique (ADE-RT) for solving the constrained optimization problems. The algorithm adaptively uses low and high crossover rate ranges and speeds the convergence with feasibility ranking mutation when all individuals are feasible. It incorporates the dynamic thresholding technique with the threshold-decreasing rate (TR). The ADE-RT using the suitable TR is compared with several well-known algorithms on benchmark problems. Experimental results show that the proposed algorithm outperforms the compared methods.

Key words and phrases: Constrained optimization problems, Adaptive differential evolution algorithm, Feasibility ranking mutation, Dynamic thresholding technique.
AMS (MOS) Subject Classifications: 68T20, 65K10.
ISSN 1814-0432, 2024, http://ijmcs.future-in-tech.net

1 Introduction

Constrained optimization problems (COPs) aim to find solutions that satisfy the constraints and give optimal objective function values. Solving COPs is challenging because the solutions must be in feasible regions defined by inequality and equality constraints. The problems are complicated when the objective functions are multimodal and high-dimensional, and the feasible regions have small fractional volumes.

The well-known Deb's feasibility rule [1] selects the preferred solutions as follows: (i) the one with a better objective function value is preferred between two feasible solutions, (ii) if one solution is infeasible and the other one is feasible, the feasible solution is preferred, and (iii) the one with a lower degree of constraint violation is preferred between two infeasible solutions.

Evolutionary algorithms (EAs) have been applied to solve COPs because they do not require derivatives and can give global solutions. Among different EAs, the differential evolution algorithm (DE) [2] is a simple and efficient population-based evolutionary method for solving continuous optimization problems. The DE algorithm consists of four operations: initialization, mutation, crossover, and selection. Its control parameters, scaling factor, crossover rate, and population size affect the search performance, and the suitable values depend on the problems. Therefore, the researchers have proposed adaptive DE algorithms for solving general and high-complexity problems.

We propose an adaptive differential evolution with feasibility ranking mutation and dynamic thresholding technique called ADE-RT. The algorithm adaptively uses low and high crossover rates to balance the local and global search and suit the problem being solved. The feasibility ranking mutation strategy ranks feasible individuals using their objective function values and positions the base vector in a mutation equation to speed up the convergence performance of the method. The algorithm incorporates the dynamic thresholding technique to handle equality constraints.

2 Literature review

Many researchers have proposed population-based methods using the improvements of the preference rule approach to handle the constraints of COPs and enhancement techniques to make search processes efficient. XU et al. [3] proposed the DE with the cooperative ranking-based mutation strategy (CRM) that sorts feasible individuals with objective function values and in-

feasible individuals with the degree of constraint violations. The algorithm calculates the probability from the ranking fractions to select the random vectors in the mutation. It selects the preferred solution with Deb's feasibility rule. The results show that the DE variants using the CRM strategy converge faster than the non-CRM-based ones on CEC2006, CEC2010, and CEC2011. Takahama and Sakai [4] proposed ε-constrained differential evolution (εDE) that relaxes the constraints to the ε level values. The algorithm uses the gradient-based mutation to adjust the trial vectors to equality constraints. The εDE can find feasible and optimal solutions for 22 out of 24 problems on CEC2006. Mohamed [5] introduced an enhanced DE, called NDE, with a thresholding technique that transforms equality constraints into inequality constraints. The algorithm uses a triangular mutation defined by the best, better, and worst individuals among random vectors to generate the mutant vector. The NDE can solve 21 out of 24 problems on CEC2006. It outperforms the compared methods on constrained engineering design and mechanical design problems regarding objective function values. Wang et al. [6] proposed a C^{2}oDE algorithm that composes the CoDE and the ε constrained methods. It generates three offspring from three mutation and crossover strategies and selects the best one using Deb's feasibility rule as a trial vector. The algorithm restarts all individuals when they are similar and infeasible to avoid stagnation in the infeasible region. The results demonstrate that the C^{2}oDE outperforms NDE on CEC2006 and CoDE on CEC2010. Xu and Zhang [7] proposed an adaptive CETDE using a two-level ε-constrained method that computes the first ε level and scales it with ranking fractions as the second ε level for each individual to select preferred solutions. The algorithm replaces target vectors with trial vectors when the population is premature. The CETDE is superior to the DE variants using Deb's feasibility rule and ε-constrained method on CEC2006 and is competitive to the compared methods on CEC2018 and five real-world problems. Zhang et al. [8] proposed an adaptive εDE that controls the ε value to relax constraints with a heuristic rule. It reduces the high ε value more quickly than the low ε value. The results show that the algorithm converges faster than εDE and DE-CRM on CEC2006 and outperforms the compared methods on fifteen constrained engineering optimization problems. Yi et al. [9] proposed IεJADE that improves JADE with ε constraint process mechanism. It decreases the ε level by an exponent function combined with the maximum constraint violation. The algorithm selects the top-best individuals by first considering the constraint violation values and then the objective function values. The results show that the IεJADE is competitive with the compared

methods on CEC2006 and CEC 2010.

3 The proposed ADE-RT algorithm

The COPs for minimization can be expressed as follows:

$$\begin{aligned} \text{Minimize} \quad & y = f(x) \\ \text{Subject to} \quad & g_r(x) \leq 0,\ r = 1, 2, \ldots, p \\ & h_s(x) = 0,\ s = p+1, \ldots, m \end{aligned} \tag{3.1}$$

where $f(x)$, $g_r(x)$, and $h_s(x)$ are the objective function, the inequality constraints, and the equality constraints, respectively. A solution $x = (x_1, \ldots, x_D)$ is feasible if x satisfies all constraints; otherwise, it is infeasible. Optimizing a COP requires balancing constraint violation and objective function values. We propose an adaptive differential evolution with feasibility ranking mutation and dynamic thresholding technique (ADE-RT). The algorithm randomizes the scaling factor in $[0.5, 0.7]$ and adaptively uses the low and high crossover rates [10]. It uses the feasibility rule incorporated with the dynamic thresholding technique to select the preferred solutions. The technique transforms equality constraints into inequality constraints with threshold value ε and decreases the ε with the threshold-decreasing rate. When all individuals are feasible at ε level, the feasibility ranking mutation strategy (FRM) ranks random vectors in the mutation equation using their objective function values to position the base vector. The ADE-RT algorithm is described as follows:

Step 1. Setting parameters:
Population size: NP, dimension: D, lower and upper bounds of population vectors: LB and UB, acceptable value: δ, threshold-decreasing rate: TR, maximum number of function evaluations: $maxFEs$, scaling factor: F, crossover rate: CR, probabilities for using low and high crossover rates: pc_1 and pc_2, and counters corresponding to pc_1 and pc_2: nc_1 and nc_2.
Step 2. Initialization:
2.1 Generate the initial population of random vectors $x_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ in the search range $[LB, UB]$ for $i = 1, 2, \ldots, NP$ and calculate their objective function values $f(x_i)$, constraint violation of inequality constraint

$$ICV_r(x_i) = \max\{0, g_r(x_i)\}, \tag{3.2}$$

constraint violation of equality constraint

$$ECV_s(x_i, \varepsilon) = \max\{0, |h_s(x_i)| - \varepsilon\}, \tag{3.3}$$

and degree of constraint violations

$$DCV(x_i, \varepsilon) = \sum_{r=1}^{p} ICV_r(x_i) + \sum_{s=p+1}^{m} ECV_s(x_i, \varepsilon). \tag{3.4}$$

Find the best vector x_{best} using the feasibility rule at the acceptable δ level by setting $\varepsilon = \delta$.

2.2 Set an initial threshold value ε as the maximum of all ICV_r and ECV_s for all x_i. Then, calculate each $DCV(x_i, \varepsilon)$.

2.3 Set $pc_1 = pc_2 = 0.5$, and $nc_1 = nc_2 = 0$.

Step 3. Mutation: For each target vector x_i, random F in $[0.5, 0.7]$, and distinct indices r_1, r_2, r_3 from $\{1, 2, \ldots, NP\}$ which are also different from i.

3.1 If there exists an individual x such that $DCV(x, \varepsilon) > 0$, then generate the mutant vector v_i by

$$v_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}). \tag{3.5}$$

3.2 Otherwise, generate v_i using the feasibility ranking mutation

$$v_i = x^*_{r_1} + F \cdot (x^*_{r_2} - x^*_{r_3}), \tag{3.6}$$

where $x^*_{r_1}$ has the best objective function value among $x_{r_1}, x_{r_2}, x_{r_3}$ and $x^*_{r_2}, x^*_{r_3}$ are remaining vectors.

Step 4. Crossover: Generate a random number t in [0, 1]. If $t \leq pc_1$, then random CR in the range of low crossover rates C_1; otherwise, random CR in the range of high crossover rates C_2. Create the trial vector u_i as follows:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } rand_j \leq CR \text{ or } j = I_{rand} \\ x_{i,j} & \text{otherwise} \end{cases} \tag{3.7}$$

where $rand_j$ is random number in $[0, 1]$ for $j = 1, 2, 3, \ldots, D$, and I_{rand} is a randomly fixed integer from $\{1, 2, \ldots, D\}$.

Step 5. Selection:

5.1 Calculate $DCV(u_i, \delta)$, $DCV(u_i, \varepsilon)$ by eq.(3.4), and $f(u_i)$.

5.2 Replace the target vectors x_i by u_i if u_i is more preferable at ε level than x_i; i.e., one of the following conditions is satisfied:

(i) If $DCV(u_i, \varepsilon) = 0$ and $DCV(x_i, \varepsilon) = 0$ and $f(u_i) < f(x_i)$,

(ii) If $DCV(u_i, \varepsilon) = 0$ and $DCV(x_i, \varepsilon) > 0$,

(iii) If $DCV(u_i, \varepsilon) > 0$ and $DCV(x_i, \varepsilon) > 0$ and $DCV(u_i, \varepsilon) < DCV(x_i, \varepsilon)$.

5.3 Update x_{best} with u_i if u_i is the preferable at the acceptable δ level than x_{best}.

Step 6. Updating parameters:
6.1 Updates pc_1 and pc_2 as follows.
6.1.1 If a better solution u_i found in the selection is generated with $CR \in C_1$, then increase $nc_1 := nc_1 + 1$; otherwise, increase $nc_2 := nc_2 + 1$.
6.1.2 If $nc_1 + nc_2 \geq 100$, then adjust the counters $nc_1 := nc_1 + 10$ and $nc_2 := nc_2 + 10$ to prevent both of them from 0.
6.1.3 Update the probabilities pc_1 and pc_2 by

$$pc_1 = 0.9 * pc_1 + 0.1 * \left(\frac{nc_1}{nc_1 + nc_2}\right) \text{ and } pc_2 = 1.0 - pc_1. \tag{3.8}$$

Reset the counters $nc_1 = nc_2 = 0$.
Step 7. Decreasing threshold value ε:
At the end of each generation, we reduce the ε using the thresholding-decreasing rate (TR). If all individuals satisfy the degree of constraint violation at the ε level, then update ε value by

$$\varepsilon := TR * \varepsilon. \tag{3.9}$$

Update each $DCV(x_i, \varepsilon)$.
Step 8. Stopping condition: Repeat all the steps $3-7$ until the stopping condition is satisfied and report x_{best}, its objective function $f(x_{best})$, and $DCV(x_{best}, \delta)$ values.

4 Experimental designs and results

We conduct a preliminary experiment to find the suitable threshold-decreasing rate and a performance comparison experiment of ADE-RT using the suitable TR with well-known algorithms on 22 benchmark problems from CEC2006 [11], including four different constraint types: linear inequality, nonlinear inequality, linear equality, and nonlinear equality constraints. The control parameters, $NP = 70, F \in [0.5, 0.7], C_1 = [0.3, 0.4]$, and $C_2 = [0.9, 1.0]$ are set for the ADE-RT. Each run is a success if x_{best} is feasible, $|h_s(x_{best})| \leq \delta = 10^{-4}$, and $f(x_{best}) - f(x^*) \leq VTR = 10^{-4}$ where x^* is the best-known solution from the reference [5]. We report the success rate (SR), the mean of function evaluations ($Mean$), and the percentage of the standard deviation ($\%SD$).

4.1 Suitable threshold-decreasing rate for ADE-RT

This experiment varies the threshold-decreasing rate $TR = 0.5$, 0.6, 0.7, 0.8, and 0.9 to compare the performance of ADE-RT with ADE-R (without dynamic thresholding technique) on five equality constraint problems $g03$, $g13$,

$g17$, $g21$, and $g23$ over 50 independent runs. We report SR and $Mean(\%SD)$ at $maxFEs = 240000$. For each problem, the $SR = 100$ is considered first. Then, the least $Mean$ value is examined. The best values are indicated in bold. Table 1 shows that the ADE-RT using $TR = 0.6, 0.7$ gives the $SR = 100$ for all test problems, and the algorithm with $TR = 0.6$ overall outperforms the others.

Table 1: Performances of ADE-RT using different TR values and ADE-R.

ADE-RT	Problem				
TR	g03	g13	g17	g21	g23
0.5	100	96	100	100	100
	58225(33.73)	49115(37.13)	47776(11.52)	71993(6.05)	115965(10.46)
0.6	100	100	100	100	100
	34649(14.59)	**41489(21.66)**	**51111(11.80)**	**78394(5.16)**	**111790(5.26)**
0.7	100	100	100	100	100
	30032(6.61)	43166(16.38)	58480(3.29)	90421(4.54)	124285(4.23)
0.8	100	100	100	96	100
	35266(5.28)	55509(15.36)	80748(3.56)	120742(6.01)	163993(4.36)
0.9	100	100	100	88	0
	56283(4.41)	89826(7.06)	146849(2.88)	199131(4.31)	-(-)
ADE-R	0	12	22	100	100
	-(-)	226936(4.23)	161848(22.07)	81629(33.47)	133772(16.37)

4.2 Performance comparison of ADE-RT and well-known algorithms

The ADE-RT using suitable $TR = 0.6$ is compared with well-known algorithms over 50 independent runs. First, we compare the success rate of ADE-RT with NDE [5], C^{2}oDE [6], adaptive εDE [8], and IεJADE [9] at $maxFEs = 240000$ as shown in Table 2. The symbols "0" and "1" denote $SR < 100$ and $SR = 100$, respectively. The results show that ADE-RT and C^{2}oDE give $SR = 100$ for all cases, while NDE, adaptive εDE, and IεJADE give $SR = 100$ for 21, 21, 16 cases out of 22 cases, respectively. Therefore, ADE-RT outperforms NDE, adaptive εDE, and IεJADE and is competitive with C^{2}oDE.

Second, we compare the performances of ADE-RT with εDE [4], DE-CRM [3], CETDE [7], and adaptive εDE [8] algorithms over 25 independent runs. The SR and $Mean(\%SD)$ at $maxFEs = 500000$ are reported in Table 3 where the results of each compared method are from the original papers. We use a two-tailed t-test at a significance level of 0.05 to compare their performances. The symbols "+" , "$\approx$", and "$-$" denote that ADE-RT performs significantly better than, similar to, and worse than the compared methods.

Table 2: Performances of ADE-RT and well-known algorithms in terms of SR.

Problem	NDE [5]	C^{2}oDE [6]	adaptive εDE [8]	IεJADE [9]	ADE-RT
g01	1	1	1	1	1
g02	0	1	0	1	1
g03	1	1	1	0	1
g04	1	1	1	1	1
g05	1	1	1	1	1
g06	1	1	1	1	1
g07	1	1	1	1	1
g08	1	1	1	1	1
g09	1	1	1	1	1
g10	1	1	1	1	1
g11	1	1	1	1	1
g12	1	1	1	1	1
g13	1	1	1	1	1
g14	1	1	1	1	1
g15	1	1	1	0	1
g16	1	1	1	1	1
g17	1	1	1	0	1
g18	1	1	1	0	1
g19	1	1	1	1	1
g21	1	1	1	0	1
g23	1	1	1	0	1
g24	1	1	1	1	1
Summarize	21	22	21	16	22

The result shows that the ADE-RT and εDE algorithms give $SR = 100$ for all cases, while DE-CRM, CETDE, and adaptive εDE methods give mean SR equal to 97.45, 99.82, 99.64, respectively. For statistical comparison, the number of cases of ADE-RT that are superior to other algorithms is 16, 14, 21, and 11 out of 22 cases, respectively. It indicates that ADE-RT outperforms the compared methods.

5 Discussion

The dynamic thresholding technique uses a threshold value ε to convert equality constraints into inequality constraints. It reduces the value with the rate TR when all individuals satisfy the feasibility rule at the current ε level at the end of a generation. Low TR rapidly decreases the ε and leads to a premature population, while high TR slows the convergence speed of the ADE-RT algorithm. We obtain the suitable $TR = 0.6$ for this study. The feasibility ranking mutation (FRM) generates a more potential mutant vector when all individuals are feasible at a ε level. It exploits the base vector with the lowest objective function value to accelerate the convergence speed. We compare the performances of ADE-RT with and without FRM on nine constraint problems at $maxFEs = 240000$ over 50 independent runs.

Table 3: Performance comparison of ADE-RT with well-known algorithms.

Problem	εDE [4]	DE-CRM [3]	CETDE [7]	Adaptive εDE [8]	ADE-RT
g01	100 (+)	100 (−)	100 (+)	100 (−)	100
	59308(1.95)	23532(6.12)	102866(0.84)	48494(4.04)	53566(2.48)
g02	100(+)	72 (+)	96 (+)	92 (+)	100
	149825(10.49)	226666(84.26)	120138(66.94)	90935(5.84)	93689(6.19)
g03	100 (+)	96 (+)	100 (+)	100 (−)	100
	89407(1.20)	63936(142.17)	89882(1.36)	27062(10.02)	35454(21.23)
g04	100 (+)	100 (−)	100 (+)	100 (−)	100
	26216(3.47)	12260(4.57)	97716(0.56)	18974(4.79)	23471(5.23)
g05	100 (+)	100 (+)	100 (+)	100 (+)	100
	97431(0.41)	40374(1.22)	99014(1.91)	44078(15.27)	29924(4.68)
g06	100 (−)	100 (−)	100 (+)	100 (+)	100
	7381(6.24)	5818(6.65)	103672(0.72)	13622(114.24)	8565(5.10)
g07	100 (+)	100 (+)	100 (+)	100 (+)	100
	74303(3.24)	81224(5.82)	120092(2.35)	55392(5.77)	49736(5.02)
g08	100 (−)	100 (−)	100 (+)	100 (≈)	100
	1139(17.04)	978(20.96)	29398(15435)	1845(180.33)	1506(15.03)
g09	100 (+)	100 (+)	100 (+)	100 (+)	100
	23121(4.98)	23696(5.02)	104290(1.06)	19464(4.39)	17797(4.26)
g10	100 (+)	100 (+)	100 (+)	100 (+)	100
	105234(6.44)	95634(5.35)	148144(3.45)	82264(5.21)	61427(4.21)
g11	100 (+)	100 (+)	100 (+)	100 (+)	100
	16420(40.02)	25996(14.81)	59632(19.80)	6773(12.99)	6137(16.84)
g12	100 (−)	100 (−)	100 (−)	100 (−)	100
	4124(19.70)	3478(23.78)	2488(23.27)	3515(25.95)	4601(28.58)
g13	100 (−)	100 (−)	100 (+)	100 (−)	100
	34738(45.94)	34012(4.65)	112548(42.90)	26953(9.62)	41933(15.09)
g14	100 (+)	100 (+)	100 (+)	100 (−)	100
	113439(3.70)	76208(5.47)	110068(2.12)	64983(3.58)	72087(4.09)
g15	100 (+)	100 (+)	100 (+)	100 (+)	100
	84216(8.49)	33716(5.66)	91396(2.09)	22498(14.34)	19414(8.67)
g16	100 (≈)	100 (−)	100 (+)	100 (+)	100
	12986(3.09)	9032(5.92)	101304(0.69)	20408(7.02)	13037(4.20)
g17	100 (+)	100 (+)	100 (+)	100 (−)	100
	98861(0.80)	51494(8.97)	164178(18.49)	43250(4.53)	49360(2.73)
g18	100 (+)	100 (+)	100 (+)	100 (+)	100
	59153(7.31)	61566(8.97)	119598(5.03)	52253(10.68)	47021(8.65)
g19	100 (+)	100 (+)	100 (+)	100 (+)	100
	356350(7.92)	145086(5.20)	143312(7.41)	144265(8.45)	132551(8.68)
g21	100 (+)	76 (+)	100 (+)	100 (−)	100
	135143(12.55)	179412(103.67)	126228(11.76)	64688(4.43)	81582(16.44)
g23	100 (+)	100 (+)	100 (+)	100 (−)	100
	200765(13.78)	187362(18.20)	187230(23.53)	104258(5.58)	111837(4.75)
g24	100 (≈)	100 (−)	100 (+)	100 (−)	100
	2952(6.26)	2546(6.48)	71696(5.56)	2845(8.58)	3092(5.26)
Mean SR	100.00	97.45	99.82	99.64	100.00
$(+, \approx, -)$	$(16, 2, 4)$	$(14, 0, 8)$	$(21, 0, 1)$	$(11, 1, 10)$	

Table 4 shows that both algorithms can solve all problems, but ADE-RT with FRM gives significantly lower $Mean$ values. Therefore, the FRM also plays a crucial role in improving convergence performance.

Table 4: Performances of ADE algorithms with and without FRM.

Prob.	ADE without FRM $SR/Mean(\%SD)$	ADE with FRM $SR/Mean(\%SD)$	Prob.	ADE without FRM $SR/Mean(\%SD)$	ADE with FRM $SR/Mean(\%SD)$
g01	100/ 63594 (2.27)	**100/ 53265 (2.14)**	g18	100 / 74631 (7.91)	**100/ 47832(8.80)**
g02	100/139212(5.48)	**100/ 94695 (7.11)**	g19	100 / 213537(4.96)	**100/130959(7.99)**
g03	100/40640(19.87)	**100/35357(18.76)**	g21	100 / 90913 (7.22)	**100/82397(16.38)**
g13	100/42571(24.38)	**100/42479(20.71)**	g23	100/140908(10.03)	**100/112008(6.43)**
g17	100/ 53570 (4.49)	**100/ 49573 (3.50)**			

6 Conclusion

We have presented an adaptive differential evolution algorithm with feasibility ranking mutation and dynamic thresholding technique for solving COPs. The algorithm enhances searchability by adaptively using low and high crossover rates. It relaxes the equality constraints with the dynamic thresholding technique using the threshold-decreasing rate and speeds up the convergence with the feasibility ranking mutation strategy. The experimental results show that our proposed algorithm can solve all CEC2006 benchmark problems and outperform the compared methods.

Acknowledgment

Wittaya Phaengthaisong thanks the Science Achievement Scholarship of Thailand (SAST) for financial support. The research on adaptive differential evolution with feasibility ranking mutation and dynamic thresholding technique for constrained optimization by Khon Kaen University has received funding support from the National Science Research and Innovation Fund (NSRF).

References

[1] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Method in Applied Mechanics and Engineering, **186**, nos. 2-4, (2000), 311–338.

[2] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization, **11**, (1997), 341–359.

[3] B. Xu, H. Zhang, M. Zhang, L. Liu, Differential evolution using cooperative ranking-based mutation operators for constrained optimization, Swarm and Evolutionary Computation, **49**, (2019), 206–219.

[4] T. Takahama, S. Sakai, Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites, Proceedings of the IEEE International Conference on Evolutionary Computation, Vancouver, BC, Canada, (2006), 1–8.

[5] A. W. Mohamed, A novel differential evolution algorithm for solving constrained engineering optimization problems, Journal of Intelligent Manufacturing, **29**, (2018), 659–692.

[6] B. C. Wang, H. X. Li, J. P. Li, Y. Wang, Composite differential evolution for constrained evolutionary optimization, IEEE Transactions on Systems, Man, and Cybernetics: Systems, **49**, no. 7, (2019), 1482–1495.

[7] B. Xu, Z. Zhang, Constrained optimization based on ensemble differential evolution and two-level-based epsilon method, IEEE Access, **8**, (2020), 213981–213997.

[8] C. Zhang, A. K. Qin, W. Shen, L. Gao, K. C. Tan, X. Li, ε-Constrained differential evolution using an adaptive ε-level control method, IEEE Transactions on Systems, Man, and Cybernetics: Systems, **52**, no. 2, (2022), 769–785.

[9] W. Yi, Z. Lin, Y. Chen, Z. Peiz, J. Lu, An enhanced adaptive differential evolution approach for constrained optimization problems, CMES-Computer Modeling in Engineering & Sciences, **136**, no. 3, (2023), 2841–2860.

[10] P. Puphasuk, J. Wetweerapong, An enhanced differential evolution algorithm with adaptation of switching crossover strategy for continuous optimization, Foundations of Computing and Decision Sciences, **45**, (2020), 97–124.

[11] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. C. Coello, K. Deb, Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization, Journal of Applied Mechanics, **41**, no. 8, (2006), 8–31.