LETTER

A Simple but Efficient Ranking-Based Differential Evolution

Jiayi LI[†], **Lin YANG**[†], **Junyan YI**[††], **Haichuan YANG**[†], **Yuki TODO**[†††], ***Nonmembers***, ***and*** **Shangce GAO**[†a)], ***Member***

SUMMARY Differential Evolution (DE) algorithm is simple and effective. Since DE has been proposed, it has been widely used to solve various complex optimization problems. To further exploit the advantages of DE, we propose a new variant of DE, termed as ranking-based differential evolution (RDE), by performing ranking on the population. Progressively better individuals in the population are used for mutation operation, thus improving the algorithm's exploitation and exploration capability. Experimental results on a number of benchmark optimization functions show that RDE significantly outperforms the original DE and performs competitively in comparison with other two state-of-the-art DE variants.

key words: *differential evolution, rank, population update*

1. Introduction

With the emergence of various complex optimization problems in the real world, meta-heuristic algorithms have become a mainstream method to solve these problems with high efficiency. Representative meta-heuristic algorithms include: genetic algorithm [1], which mimics the evolutionary mechanism of biological populations, particle swarm optimization [2], [3], which is designed to simulate the predatory behavior of bird flocks, differential evolution (DE) [4], [5], which uses a differential mutation strategy, and spherical evolution [6], [7], which uses a spherical search strategy.

DE algorithm was proposed by Store et al. in 1997 [4], and has the characteristics of simplicity and effectiveness in comparison with other evolutionary algorithms [8]. However, it suffers from the issues of convergence premature and local optima trapping. Therefore, many variants of DE algorithm have been proposed to improve its performance. Two representative ones are JADE [9] which uses an external archive to restore promising solutions, and CJADE [10] which uses multiple chaotic local search to enhance its exploitation ability. Both algorithms dynamically adjust the values of mutation factor and crossover probability by different methods to improve the performance of DE algorithm. Although these improvements are significant, they are sophisticated and not easily to be implemented and adjusted for other kinds of optimization tasks, thus limiting their flexibility and applicability.

In order to further enhance DE's performance, in this study, we attempt to improve it from the perspective of population structure. It is widely accepted that the population structure directly determines the information communication among individuals, thus making influence on the search performance of the algorithm indirectly [11]. Various population structures have been used in several meta-heuristics, including hierarchical structure [12], island-based multiple population structure [13], and scale-free-network based structure [14]. Although these sophisticatedly designed population structures are effective, they are not straightforward and difficult to be applied on DE. Innovatively, this paper proposes a novel bi-population structure for DE by ranking individuals according to their fitness. The whole population is divided into two sub-populations: one is used to store better individuals, while the other stores worse individuals. Thereafter, better individuals are selected for the mutation strategy, and then crossed with worse individuals to produce a new population. By this method, the balance between global exploration and local exploitation of DE can be realized. Extensive experiments are conducted based on thirty IEEE Congress on Evolutionary Computation (CEC) 2017 benchmark optimization functions to verify the effectiveness of the newly proposed ranking-based differential evolution (RDE) algorithm.

The contributions of this paper are as follows: 1) To the best of our knowledge, this paper for the first time presents an improved DE algorithm based on ranking from the aspect of population structure. 2) Experimental results show that RDE outperforms its peers in terms of solution quality, and thus it opens the door to the research that applying such ranking-based bi-population structure to other meta-heuristic algorithms.

2. Differential Evolution

First, we briefly introduce the DE algorithm. The DE algorithm has three important strategies, i.e., mutation, crossover and selection. Mutation and crossover are used to generate new individuals, and selection is used to retain better individuals to the next population. DE uses a simple and efficient mutation strategy called differential mutation, and the

Manuscript received May 26, 2021.
Manuscript revised August 15, 2021.
Manuscript publicized October 5, 2021.

[†]The authors are with Faculty of Engineering, University of Toyama, Toyama-shi, 930–8555 Japan.
[††]The author is with Beijing University of Civil Engineering and Architecture, Beijing, 100044 China.
[†††]The author is with Faculty of Electrical, Information and Communication Engineering, Kanazawa University, Kanazawa-shi, 920–1192 Japan.
a) E-mail: gaosc@eng.u-toyama.ac.jp

DOI: 10.1587/transinf.2021EDL8053

formula is as follows:

$$V_i(k) = X_{r1}(k) + F \times (X_{r2}(k) - X_{r3}(k)), \quad i, r1, r2, r3 \in [1, N] \tag{1}$$

where $V_i(k)$ denotes the ith individual after the kth generation of population mutation. N is the number of individuals in the population. F is the scaling factor in the mutation strategy, which is usually set to a constant value of 0.5 [15]. $r1$, $r2$, $r3$ are random integers between 1 to N that are different from each other.

The crossover strategy of the DE algorithm uses individuals in the mutated population $V(k)$ to crossover with the individuals in the original population $X_i(k)$ which is the ith individual in the kth generation, formulated as follows:

$$U_{i,j}(k) = \begin{cases} V_{i,j}(k), & rand(0,1) < CR \ or \ j = j_{rand} \\ X_{i,j}(k), & otherwise \end{cases} \tag{2}$$

where $j = 1, 2, \ldots, D$, CR is the crossover probability, which determines the extent to which elements of the population individuals are replaced by elements of the mutated individuals, and CR is set to a constant value of 0.9. D is the dimension size of the optimization problem. j_{rand} is a random integer between 1 and D.

After executing the mutation and crossover strategies, the DE algorithm calculates the fitness of all individuals in the population $U(k)$. Using a greedy selection strategy, it compares the individuals in the original population $X(k)$ and selects better individual to be survived into the next generation. The implementation formula is as follows:

$$X_i(k+1) = \begin{cases} U_i(k), & f(U_i(k)) < f(X_i(k)) \\ X_i(k), & otherwise \end{cases} \tag{3}$$

where $f()$ represents the function to calculate the fitness of an individual.

3. Ranking-Based DE

This paper proposes a new RDE algorithm based on the DE algorithm. The difference in flowchart between RDE and DE is illustrated in Fig. 1. Compared with DE, RDE has an additional operation of population sorting to obtain better and worse individuals. The RDE algorithm uses better individuals for mutation, and worse individuals for crossover. It ensures the diversity of the population during iteration and significantly enhances the convergence of the algorithm.

In RDE, we first sort the population $X(k)$ from best to worst by computing fitness to obtain $X^{sort}(k)$. We define the first half of $X^{sort}(k)$ as $X^{good}(k)$ and the second half as $X^{bad}(k)$, formulated as:

$$\begin{cases} X_i^{good}(k) = X_{i'}^{sort}(k) \\ X_i^{bad}(k) = X_{i''}^{sort}(k) \end{cases} \tag{4}$$

where $i' = 1, 2, \ldots, \lceil \frac{N}{2} \rceil$; $i'' = \lceil \frac{N}{2} \rceil + 1, \lceil \frac{N}{2} \rceil + 2, \ldots, N$.

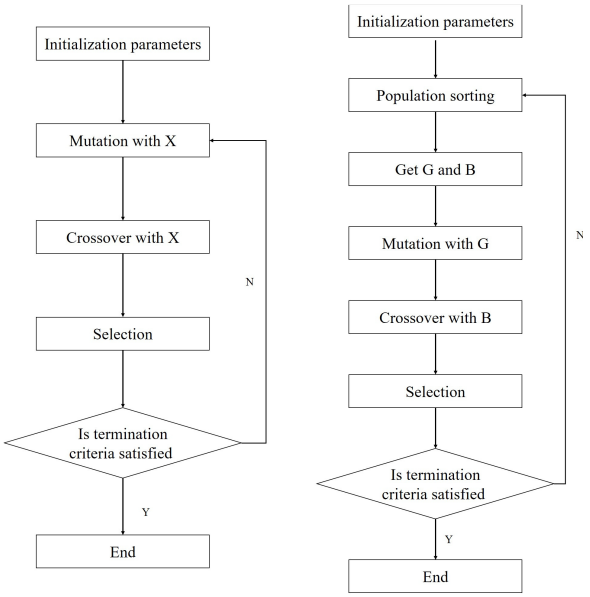

Fig. 1 (a) The DE structured diagram. (b) The RDE structured diagram.

Then, we randomly select a total of $\lceil \frac{N}{2} \rceil$ individuals from $X^{good}(k)$ and $X^{bad}(k)$ as the population $G(k)$, while we combine the individuals not selected as the population $B(k)$. The formulas are as follows:

$$G_i(k) = \begin{cases} X_i^{good}(k), & r < p \\ X_i^{bad}(k), & r \geq p \end{cases} \text{ for } i \in \left[1, \left\lceil \frac{N}{2} \right\rceil\right] \tag{5}$$

$$B_i(k) = X_{i'}(k), \ X_{i'}(k) \notin G(k), \ i \in \left[1, \left\lceil \frac{N}{2} \right\rceil\right], i' \in [1, N] \tag{6}$$

where p is a value used to regulate the number of individuals in $X^{good}(k)$ and $X^{bad}(k)$ owned by the population of $G(k)$. To obtain a $G(k)$ consisting of better individuals, we take $p \times \lceil \frac{N}{2} \rceil$ random individuals from $X^{good}(k)$ and $(1-p) \times \lceil \frac{N}{2} \rceil$ random individuals from $X^{bad}(k)$, which adds up to a total of $\lceil \frac{N}{2} \rceil$ individuals to form $G(k)$. The formula for calculating p is as follows:

$$p = 0.5 \cdot \left(1 + \frac{T}{T_{max}}\right) \tag{7}$$

where T represents the current number of evaluations of the algorithm and T_{max} represents the predefined maximum number of evaluations of the algorithm. Clearly, $p \in [0.5, 1]$. As the number of iterations of the algorithm increases, the number of individuals taken from $X^{good}(k)$ in the population $G(k)$ gradually becomes larger, and finally $G(k)$ is constructed by individuals all from $X^{good}(k)$. Thus, better individuals are gradually stored in $G(k)$, while worse ones are archived in $B(k)$.

The mutation formula of the RDE algorithm is implemented only on $G(k)$, shown as:

$$V_i(k) = G_{r1}(k) + F \times (G_{r2}(k) - G_{r3}(k)), \tag{8}$$

where $i \in \left[1, \left\lceil \frac{N}{2} \right\rceil\right]$, $V_i(k)$ is the ith individual after mutation. $r1, r2, r3$ are random integers with values from 1 to $\left\lceil \frac{N}{2} \right\rceil$, respectively.

From Eq. (8), we can find that in the early search stage some worse individuals taken from $X^{bad}(k)$ can participate in the mutation, which is beneficial to maintain the population diversity and jump out of the local optima to some extent. On the other hand, in the late stage of the search RDE is inclined to use more better individuals to perform mutation, thus significantly improving its exploitation ability and accelerating the convergence speed. Thus, the balance of exploitation and exploration in RDE is realized by automatically adjusting the p value.

In the crossover operation, we use the mutated individuals $V(k)$ from the $G(k)$ sub-population to crossover with those individuals in the sub-population $B(k)$. The formula is as follows:

$$U_{i,j}(k) = \begin{cases} V_{i,j}(k), & rand(0,1) < CR \ or \ j = j_{rand} \\ B_{i,j}(k), & otherwise \end{cases} \tag{9}$$

where $i = 1, 2, \ldots, \left\lceil \frac{N}{2} \right\rceil$, $j = 1, 2, \ldots, D$, $U(k)$ represents the new generation population generated by the crossover operation. It is worth pointing out that the participation of $B(k)$ in the crossover can not only balance the fitness difference between two sub-populations, but also improve the population diversity by fully using the information of worse individuals and thus enhance the exploration ability of the algorithm to avoid falling into local optimum too quickly.

It is notable that there are only $\left\lceil \frac{N}{2} \right\rceil$ number of individuals in the new population generated in the RDE algorithm. Thus the selection strategy of RDE is also different from that of DE. We retain the best individuals in the original population that are in the good sub-population $X^{good}(k)$. The RDE algorithm uses a greedy selection strategy to compare the new population U with the individuals in the original population $X^{bad}(k)$ and select better individuals as the new individuals. The implementation formula is as follows:

$$X_i(k+1) = \begin{cases} U_{i'}(k), & f(U_{i'}(k)) < f(X_i(k)), X_i(k) \in X^{bad}(k) \\ X_i(k), & otherwise \end{cases} \tag{10}$$

where $i' \in \left[1, \left\lceil \frac{N}{2} \right\rceil\right]$ represents the number of population $U(k)$. With this selection strategy, the RDE algorithm gives better individuals more chances to participate in the next iteration and will gradually replace worse individuals. Compared with DE, RDE can achieve a better balance between global exploration and local exploitation.

4. Experimental Results

To test the capability of the RDE algorithm, we implemented experiments based on 30 benchmark optimization

Table 1 Experiment results on CEC2017.

	RDE		DE			JADE			CJADE		
	mean	std	mean	std		mean	std		mean	std	
F1	1.00E-14	1.07E-14	**6.41E-15**	9.97E-15	−	1.50E-14	4.41E-15	+	1.62E-14	5.70E-15	+
F2	6.49E+16	3.43E+17	6.48E+06	2.00E+07	−	1.79E+03	1.28E+04	−	**1.01E-12**	1.81E-12	−
F3	**3.75E-02**	1.23E-01	2.88E+01	2.48E+01	+	5.06E+03	1.32E+04	−	9.64E+03	1.56E+04	−
F4	6.80E+01	2.49E+01	5.00E+01	2.14E+01	−	4.76E+01	2.48E+01	−	**1.23E+01**	2.27E+01	−
F5	1.32E+02	6.30E+01	1.77E+02	1.14E+01	+	**2.66E+01**	4.26E+00	−	2.70E+01	4.82E+00	−
F6	2.68E-09	1.92E-08	2.38E-08	3.94E-08	+	**1.14E-13**	0.00E+00	≈	1.29E-13	5.09E-14	+
F7	1.87E+02	3.80E+01	2.10E+02	1.01E+01	+	5.46E+01	3.55E+00	−	**5.31E+01**	4.40E+00	−
F8	1.23E+02	6.71E+01	1.81E+02	1.01E+01	≈	**2.58E+01**	4.73E+00	−	2.66E+01	4.13E+00	−
F9	**0.00E+00**	0.00E+00	**0.00E+00**	0.00E+00	≈	3.51E-03	1.76E-02	+	1.42E-02	6.64E-02	≈
F10	6.93E+03	3.01E+02	7.01E+03	2.92E+02	+	**1.92E+03**	2.52E+02	−	1.93E+03	1.99E+02	−
F11	**1.31E+01**	1.40E+01	5.27E+01	1.42E+01	+	2.89E+01	2.57E+01	+	3.12E+01	2.40E+01	+
F12	7.82E+03	5.36E+03	6.21E+03	3.84E+03	−	1.15E+03	3.94E+02	−	**1.11E+03**	6.05E+02	−
F13	5.07E+01	2.74E+01	8.11E+01	8.87E+00	+	**4.15E+01**	1.64E+01	≈	2.04E+02	1.17E+03	≈
F14	**1.56E+01**	1.67E+01	6.31E+01	4.62E+00	+	1.90E+03	5.97E+03	+	1.88E+03	4.82E+03	+
F15	**7.82E+00**	2.16E+00	3.82E+01	5.60E+00	+	2.79E+02	1.34E+03	+	1.54E+02	6.56E+02	+
F16	5.35E+02	3.25E+02	1.10E+03	3.83E+02	+	**4.16E+02**	1.47E+02	−	4.48E+02	1.45E+02	−
F17	**3.34E+01**	1.75E+01	7.35E+01	7.90E+00	+	7.15E+01	2.43E+01	+	7.22E+01	2.84E+01	+
F18	**2.14E+01**	9.93E+00	3.62E+01	4.47E+00	+	1.89E+04	6.45E+04	+	3.59E+03	1.81E+04	+
F19	**4.50E+00**	1.38E+00	1.66E+01	5.36E+00	+	4.19E+02	1.64E+03	+	4.15E+02	2.86E+03	+
F20	**7.94E+00**	2.57E+01	5.25E+01	6.12E+01	+	1.12E+02	5.27E+01	+	1.23E+02	5.68E+01	+
F21	3.14E+02	6.39E+01	3.70E+02	9.59E+00	+	**2.26E+02**	3.97E+00	−	2.26E+02	4.52E+00	−
F22	1.00E+02	1.00E-13	1.00E+02	1.17E-13	≈	1.00E+02	1.00E-13	≈	1.00E+02	1.00E-13	≈
F23	3.74E+02	4.62E+01	5.22E+02	1.22E+01	+	3.74E+02	5.14E+00	+	**3.71E+02**	6.06E+00	+
F24	4.49E+02	3.74E+01	5.95E+02	8.96E+00	+	4.40E+02	4.72E+00	≈	**4.38E+02**	4.65E+00	≈
F25	3.87E+02	4.73E-01	**3.87E+02**	3.30E-02	−	3.87E+02	5.35E-01	+	3.88E+02	7.25E+00	+
F26	**9.95E+02**	9.47E+01	2.32E+03	3.23E+02	+	1.16E+03	1.44E+02	+	1.17E+03	6.65E+01	+
F27	4.83E+02	1.01E+01	**4.77E+02**	9.08E+00	≈	5.04E+02	6.64E+00	+	5.04E+02	6.30E+00	+
F28	**3.27E+02**	4.93E+01	3.30E+02	4.87E+01	+	3.42E+02	5.49E+01	+	3.30E+02	5.23E+01	≈
F29	**4.18E+02**	2.85E+01	6.21E+02	1.22E+02	+	4.81E+02	2.52E+01	+	4.76E+02	2.29E+01	+
F30	2.32E+03	5.43E+02	2.14E+03	8.17E+01	≈	**2.09E+03**	1.31E+02	−	2.21E+03	1.54E+02	≈
W/T/L	−/−/−		20/5/5			15/4/11			14/6/10		

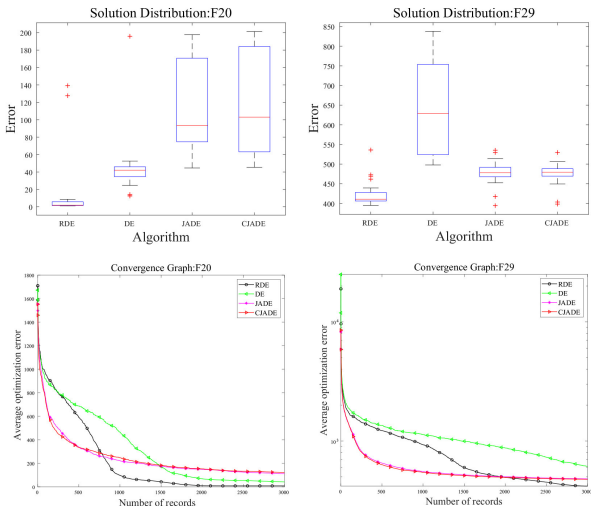

Fig. 2 Convergence and box-plot graphs of (a) F20 and (b) F29.

functions of IEEE CEC2017, including four types of problems: unimodal functions (F1–F3), simple multimodal functions (F4–F10), hybrid functions (F11–F20), and combined functions (F21–F30). The maximum function evaluating number we set is $10000D$. The dimension D is 30, the population size is 100, and each function is run 51 times independently. DE [15] with $F = 0.5$ and $CR = 0.9$, JADE [9] and CJADE [10] are used as performance competitors.

Table 1 summarizes the experimental results of solutions error in terms of mean and standard deviation (std) values, where the best value among all comparative algorithms for each function is shown in bold. Additionally, we also use Wilcoxon statistical test with a significance level of $\alpha = 0.05$ to detect the difference among algorithms [16]. The W/T/L values indicate the number of functions on which the RDE algorithm performs significantly better (marked with +), tied (marked with $\approx$) or worse (marked with $-$) than its counterpart, respectively. From the table, we can clearly find that RDE significantly outperforms DE, and performs competitively with JADE and CJADE.

Furthermore, Fig. 2 shows the box-and-whisker plots and convergence graphs obtained by the tested algorithms. Two typical functions F20 and F29 are used as examples to show the distribution of solutions and the convergence characteristics of RDE, DE, JADE and CJADE. From it, it is obvious that RDE has a faster convergence speed in the later search phase and can find better solutions than its peers.

5. Conclusions

In this paper, we propose a simple but effective ranking-based DE (RDE) algorithm. A bi-population is used by a ranking strategy, and the sub-population that stores better individuals generally performs mutation, while the other one that archives worse individuals are used to implement crossover. The resultant RDE shows great potential in solving complex optimization problems. In the future, we plan to use RDE to solve some real-world problems, such as neural network learning [17], parameter estimation [18], etc.

References

[1] J.H. Holland, "Genetic algorithms," Scientific American, vol.267, no.1, pp.66–73, 1992.
[2] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," Swarm Intelligence, vol.1, no.1, pp.33–57, 2007.
[3] S. Song, J. Ji, X. Chen, S. Gao, Z. Tang, and Y. Todo, "Adoption of an improved PSO to explore a compound multi-objective energy function in protein structure prediction," Applied Soft Computing, vol.72, pp.539–551, 2018.
[4] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," Journal of Global Optimization, vol.11, no.4, pp.341–359, 1997.
[5] J. Sun, S. Gao, H. Dai, J. Cheng, M. Zhou, and J. Wang, "Bi-objective elite differential evolution for multivalued logic networks," IEEE Trans. Cybern., vol.50, no.1, pp.233–246, 2020.
[6] D. Tang, "Spherical evolution for solving continuous optimization problems," Applied Soft Computing, vol.81, 105499, 2019.
[7] H. Yang, S. Gao, R.-L. Wang, and Y. Todo, "A ladder spherical evolution search algorithm," IEICE Trans. Inf. & Syst., vol.E104-D, no.3, pp.461–464, March 2021.
[8] Bilal, M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, "Differential evolution: A review of more than two decades of research," Engineering Applications of Artificial Intelligence, vol.90, 103479, 2020.
[9] J. Zhang and A.C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," IEEE Trans. Evol. Comput., vol.13, no.5, pp.945–958, 2009.
[10] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, and M. Zhou, "Chaotic local search-based differential evolution algorithms for optimization," IEEE Trans. Syst. Man Cybern. Syst., vol.51, no.6, pp.3954–3967, 2021.
[11] J.L. Payne, M. Giacobini, and J.H. Moore, "Complex and dynamic population structures: Synthesis, open questions, and future directions," Soft Computing, vol.17, no.7, pp.1109–1120, 2013.
[12] Y. Wang, Y. Yu, S. Gao, H. Pan, and G. Yang, "A hierarchical gravitational search algorithm with an effective gravitational constant," Swarm and Evolutionary Computation, vol.46, pp.118–139, 2019.
[13] Y. Wang, S. Gao, M. Zhou, and Y. Yu, "A multi-layered gravitational search algorithm for function optimization and real-world problems," IEEE/CAA Journal of Automatica Sinica, vol.8, no.1, pp.94–109, 2021.
[14] J. Ji, S. Song, C. Tang, S. Gao, Z. Tang, and Y. Todo, "An artificial bee colony algorithm search guided by scale-free networks," Information Sciences, vol.473, pp.142–165, 2019.
[15] S. Das and P.N. Suganthan, "Differential evolution: A survey of the state-of-the-art," IEEE Trans. Evol. Comput., vol.15, no.1, pp.4–31, 2010.
[16] J. Carrasco, S. García, M. Rueda, S. Das, and F. Herrera, "Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review," Swarm and Evolutionary Computation, vol.54, 100665, 2020.
[17] S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neural model with effective learning algorithms for classification, approximation, and prediction," IEEE Trans. Neural Netw. Learn. Syst., vol.30, no.2, pp.601–614, 2019.
[18] S. Gao, K. Wang, S. Tao, T. Jin, H. Dai, and J. Cheng, "A state-of-the-art differential evolution algorithm for parameter estimation of solar photovoltaic models," Energy Conversion and Management, vol.230, 113784, 2021.