International Journal of Innovative
Computing, Information and Control
Volume **16**, Number **6**, December **2020**

ICIC International ©2020 ISSN 1349-4198
pp. 1845–1862

AN IMPROVED MULTI-VERSE OPTIMIZER ALGORITHM FOR MULTI-SOURCE ALLOCATION PROBLEM

Ruixing Song[1,2], Xuewen Zeng[1,2] and Rui Han[1,*]

[1]National Network New Media Engineering Research Center
Institute of Acoustics, Chinese Academy of Sciences
No. 21, North 4th Ring Road, Haidian District, Beijing 100190, P. R. China
{ songrx; zengxw }@dsp.ac.cn; *Corresponding author: hanr@dsp.ac.cn

[2]School of Electronic, Electrical and Communication Engineering
University of Chinese Academy of Sciences
No. 19(A), Yuquan Road, Shijingshan District, Beijing 100049, P. R. China

Received May 2020; revised September 2020

Abstract. *Multi-verse optimizer (MVO) algorithm is a nature-inspired algorithm for solving single-objective optimization problems. MVO algorithm has many advantages, including few parameters, excellent performance, fast convergence, and low resource consumption. However, it is easy to fall into the local optimum condition and its fineness is not enough during processing multi-resource allocation in Web-based media presentation. Herein, this work proposes an improved MVO (abbreviated as RISEMVO) algorithm. The wormhole existence probability and the travelling distance rate were modified to improve the exploitation capability, and the strategy of revolving around the best universe was added to improve the exploitation and exploration capabilities. Moreover, the jumping of local optimal strategy was added. In order to reflect the performance differences more appropriately between RISEMVO and MVO algorithms, we firstly tested them with test functions used by the original authors of MVO algorithm. RISEMVO algorithm performs best in 29 test functions compared with standard MVO algorithm and other four commonly used algorithms. We applied RISEMVO algorithm to multi-resource allocation in Web-based media presentation, which enables the maximum utilization of the system and outperforms other 5 algorithms. These results demonstrate the advantages of RISEMVO algorithm in most test functions and in solving the multi-resource allocation problem.*
Keywords: Improved multi-verse optimizer algorithm, Meta heuristic optimization, Nonlinear convergence factor, Random variation, Web, Multi-resource allocation problem

1. **Introduction.** Streaming media traffic has accounted for 75% of current network traffic; moreover, videos will consume 79% of the total mobile traffic by 2022 [1,2]. Among the diverse video transmission solutions, Web-based media presentation has become a prevalent solution to media transmission. However, as the equipment resources for media presentation are limited, the multiple operations run in parallel and consume various resources, and herein, inappropriate resource allocation schemes will cause the interruption of media presentation or will waste unnecessary resources [3].

The resource allocation problem in media presentation can be attributed to the multi-resource allocation problem with successive dependencies and priority settings, which is an important problem limiting the performance in many fields [4-6]. To solve the problem, it is abstractly quantified as an optimization problem model, followed by selecting appropriate problem solution and employing intelligent optimization algorithms. Among

DOI: 10.24507/ijicic.16.06.1845

the employed intelligent optimization algorithms, heuristic optimization algorithms have become a solution tendency in this field [7,8].

Multi-verse optimizer (MVO) algorithm is a population-based heuristic optimization algorithm proposed by Mirjalili et al. in 2016 [9,10]. MVO algorithm has the advantages of few parameters, excellent performance, fast convergence, and low resource consumption, which makes MVO algorithm capable of solving optimization problems in complex situations like multi-resource allocation problem. The main inspirations of MVO algorithm are based on three concepts in cosmology: white holes, black holes, and wormholes. The mathematical models of these three concepts are developed to perform exploration, exploitation and local search, respectively. The exploration and exploitation are two phases of typical search process of meta-heuristic optimization algorithm. In exploration phase, search agents investigate the search space as widely as possible to obtain the promising region. While in exploitation phase, search is carried out in the local region obtained by exploration phase to find the global optimum solution. However, MVO algorithm still has some limitations, such as being easy to fall into local optimization and insufficient fineness. These will induce failure to gain maximized resource efficiency during processing multi-resource allocation in Web-based media presentation.

To overcome the disadvantages and improve the accuracy of MVO algorithm, an improved MVO algorithm (abbreviated as RISEMVO) is proposed in this work. Firstly, the wormhole existence probability (WEP) and the travelling distance rate (TDR) were modified to improve the exploitation capability. Then the strategy of revolving around the best universe was added to improve the exploitation and exploration capabilities. Finally, the jumping of local optimal strategy was added. Based on these improvements, the proposed algorithm, RISEMVO algorithm, was tested with CEC (Congress on Evolutionary Computation) benchmark test functions (which was used in the paper of standard MVO algorithm) [11,12], and used to solve the multi-resource allocation problem [13-16] with successive dependencies and priority settings. The results show that RISEMVO algorithm performs better than the standard MVO algorithm.

The rest of this paper is organized as follows. Section 2 outlines the standard MVO algorithm. Section 3 presents the improved MVO algorithm (RISEMVO algorithm). Section 4 provides the experimental results and the performance analysis of RISEMVO algorithm and other related algorithms in solving the CEC benchmark test functions. Section 5 shows the application of RISEMVO algorithm in solving the Web device multi-resource allocation problem, while the corresponding experimental parameters and results are shown in Section 6. In the end, Section 7 presents the conclusions and future expectations.

2. **Multi-Verse Optimizer (MVO) Algorithm.**

The mathematical model of MVO simulates the big bang theory and the multi-verse theory, which utilizes the concepts of the white holes and the black holes to explore search space, and the wormholes to exploit the search space [17]. MVO algorithm assumes that each solution is analogous to a universe and each variable in the solution is an object in that universe [9]. In addition, MVO algorithm assigns an inflation rate to each solution, and the inflation rate is proportional to the corresponding fitness function (e.g., in Section 4, a test function is a kind of fitness function) value of the solution, and the universe with the highest inflation rate is defined as the best universe [9].

During optimization, the following rules are applied to the universes of MVO algorithm [9]. 1) The higher inflation rate, the higher probability of having white holes whereas the lower probability of having black holes. 2) Universes with higher inflation rate tend to send more objects through white holes, while universes with lower inflation rate tend to

receive more objects through black holes. 3) The objects in all universes may randomly move towards the best universe via wormholes regardless of the inflation rate.

According to the inflation rate, MVO algorithm selects a black hole with a certain probability (the current universe inflation rate is compared with a random number in $[0, 1]$, if smaller, select it), uses roulette to select a universe as a white hole, and passes the white hole coordinates to the black hole. Besides, MVO algorithm selects the universe with a probability for wormhole to cross the best universe for nearby exploration and exploitation, and updates the location of the selected universe. The universe update equation is presented in Equation (1):

$$x_j^i = \begin{cases} \begin{cases} best_{x_j} + TDR \times ((ub_j - lb_j) \times r_4 + lb_j) & r_3 < 0.5 \\ best_{x_j} - TDR \times ((ub_j - lb_j) \times r_4 + lb_j) & r_3 \geq 0.5 \end{cases} & r_2 < WEP \\ x_j^i & r_2 \geq WEP \end{cases} \tag{1}$$

where x_j^i is the j^{th} parameter of i^{th} universe, $best_{x_j}$ indicates the j^{th} parameter of the best universe formed so far; TDR and WEP are the travelling distance rate (TDR) and the wormhole existence probability (WEP), respectively; lb_j shows the lower bound of the j^{th} parameter, ub_j is the upper bound of the j^{th} parameter, and r_2, r_3, r_4 are random numbers in $[0, 1]$.

There are two main coefficients herein: WEP and TDR. WEP is for defining the probability of wormhole's existence in universes. This probability is required to increase linearly over the iterations in order to emphasize exploitation during the progress of optimization process. TDR is a factor to define the distance rate (variation) that an object can be teleported by a wormhole around the best universe obtained so far. In contrast to WEP, TDR is decreased over the iterations to enable more precise exploitation and local search around the best obtained universe. The adaptive equations for both coefficients are determined by Equation (2) and Equation (3):

$$TDR = 1 - \left(\frac{t_n}{t_{\max}}\right)^{\frac{1}{6}} \tag{2}$$

where t_n is the current iteration, and $t_{\max}$ is the maximum iteration.

$$WEP = WEP_{\min} + \frac{t_n}{t_{\max}} \times (WEP_{\max} - WEP_{\min}) \tag{3}$$

where $WEP_{\min}$ is the minimum of WEP, and $WEP_{\max}$ is its maximum.

3. Improved Multi-Verse Optimizer (RISEMVO) Algorithm.

Though MVO algorithm has many advantages, it also has the disadvantages including low exploitation and convergence speeds, and it may fall into local optimization [18,19]. To handle these deficiencies, improvements from four aspects are proposed to optimize the original model. The details of these improvements are explicitly in this part. At the end of this section, 3.5 shows the process of RISEMVO algorithm.

3.1. Revised nonlinear convergence coefficient.

The setting of TDR is modified to improve the exploitation ability of the algorithm here. MVO algorithm completes the optimization process by black-white hole material exchange and wormhole migration. The ordinary universe explores around the current best universe with TDR. Hence, TDR determines the speed and the depth of exploration and exploitation of a universe around the best universe. A larger TDR value at the beginning helps the algorithm to explore the global, and the subsequent reduction of TDR value helps the algorithm exploit around the current best universe.

For in-depth exploitation, the setting of *TDR* greatly impacts the exploitation ability of the algorithm. However, it can be seen from Equation (2) that the current *TDR* converges nonlinearly with increasing number of iterations. The value of *TDR* decreases from a maximum of 0.6 to a minimum of 0 in a concave shape, and its magnitude changes slightly when $t_{\max} = 500$ in [9]. Hence, it is important to improve its decrease way, which is modified as following Equation (4):

$$TDR = 2 \times \left(1 - \left(\frac{t_n}{t_{\max}}\right)^{\frac{1}{t_n}}\right) \tag{4}$$

Compared with original *TDR* equation, we alter the index of *TDR* in improved algorithm, and modify the original parameter 1/6 to $1/t_n$, which slow down the convergence rate and greatly increase the maximum value of *TDR* to 2 after multiple tests on different test functions. As the *TDR* is sufficiently large at the beginning, the improved algorithm is able to perform a large-scale global search. Meanwhile, the improved algorithm develops fine-grained depth near the best universe, which in theory can effectively increase the performance optimization possibility of the algorithm.

3.2. **Revised *WEP*.** To reduce the resource consumption and the algorithm complexity caused by the algorithm updating of *WEP*, we revise the *WEP* value. From Equation (3), it is clear that the *WEP* of MVO algorithm increases linearly with increasing iterations. According to the comparison of probability of *WEP* and random numbers, the algorithm chooses whether to cross the vicinity of the best universe for exploration and exploitation or not. According to previous research, the *WEP* value has little effect on the current best universe [19].

Referring to [19], the *WEP* value in the improved wormhole existence mechanism is selected to be 0.5. This setting stems from the idea of competitive confrontation in the algorithm optimization, in which process two individuals are randomly selected to compete, and the strong one enters while the weak one gets out. In order to further verify the reliability of the value of 0.5, we also comprehensively compared the parameters between 0.1∼0.9 through multiple experiments, and 0.5 is the most preferred value. In addition, for all the universes to be selected, half of the universes will locate near the best universe, which become candidates for white holes and the best universes, while the remaining universes will be eliminated and enter into another mechanism for processing before competition. Meanwhile, all universes have a fixed probability of evolution. Based on the *WEP* revisions, the improved algorithm will perform better in increasing the convergence speed of the algorithm at the beginning and avoiding the risk of falling into the local optimum phenomenon for universe gathers near the best universe in the later stage of iteration.

3.3. **Improvements in exploration methods.** New exploration methods are proposed to increase the exploration and exploitation of the universe around the best universe. At the same time, the distance between the current universe and the best universe is used to revolve the current universe, and increase the possibility of obtaining the optimal solution.

Though *TDR* and *WEP* are modified, in the condition of the current universe traveling through the wormhole to the vicinity of the best universe, the exploration and exploitation are only carried out in the front and rear positions of the best universe in MVO. However, the left and the right positions of the best universe can also be explored by slight deviation and be exploited in depth. Therefore, an additional *TDR* coefficient, TDR_2, is added which represents the travelling distance rate in left and right directions. TDR_2 is

determined by Equation (5):

$$TDR_2 = \frac{1}{2} \times \left(1 - e^{-\left(\frac{t_n}{t_{\max}}\right)^3}\right) \tag{5}$$

where t_n is the current iteration, and $t_{\max}$ is the maximum iterations.

Moreover, during the universe traversing through the wormhole and exploring in forward and backward directions, standard MVO algorithm ignores the current position information of the universe except the best universe. This wastes the information obtained by the cosmic group in the optimization process. Herein, we consider the relationship between the best universe and the current universe, and introduce a random factor to replace the original random equation, i.e., replacing $((ub_j - lb_j) \times r_4 + lb_j)$ by $\left(\left(best_{x_j} \times 2 \times r_4 - x_j^i\right) \times r_5\right)$ in Equation (1), where r_4 and r_5 are random numbers in $[0, 1]$. Thus, the way in which the universe updates its position during crossing through a wormhole follows Equation (6):

$$x_j^i = \begin{cases} best_{x_j} \times TDR_2 + TDR \times \left(\left(best_{x_j} \times 2 \times r_4 - x_j^i\right) \times r_5\right) & r_3 < 0.5 \\ best_{x_j} \times TDR_2 - TDR \times \left(\left(best_{x_j} \times 2 \times r_4 - x_j^i\right) \times r_5\right) & r_3 \geq 0.5 \end{cases} \tag{6}$$

Apart from explosions, black holes, white holes, and wormholes simultaneously cross in the universe, so all currently known nebula objects have position rotation and revolve around other objects. Through constant rotation and revolution, nebula objects like Earth are constantly updating their locations, and the probability of the best location appearing during the update process is greatly increased.

Drawing on this phenomenon, we improve the current location update method of the universe and increase the choice of the universe revolution. Before the appearance of the wormholes, the distance between the current universe and the best universe is taken as the revolution radius, and the revolution is centered on the best universe, which can be expressed by the following equation:

$$\begin{aligned} x^i(t_n + 1) = &\left|x^i(t_n) - best_{x(t_n)}\right| \times e^{\left(1 - r \times \frac{t_n}{t_{\max}}\right)} \times \cos\left(2\pi \times r \times \frac{t_n}{t_{\max}}\right) \\ &+ TDR_2 \times best_{x(t_n)} \end{aligned} \tag{7}$$

where r is a random number in $[0, 1]$, x^i is a universe that was not selected to be traversed by a wormhole, and $best_x$ is the best universe at the current iteration. With these improvements, the modified algorithm increases the way of the universe exploiting and exploring, makes full use of the information carried by the current universe, and obtains the optimal solution more effectively.

3.4. **Strategy of jumping out of local optimal solution.** In order to avoid falling into a locally optimal situation, we introduce multiple random jump perturbance on the optimal position of the universe at each iteration. The position of the j^{th} parameter of the new universe after the jump perturbance is determined to be $((ub_j - lb_j) \times r_4 + lb_j)$, where r_4 is a random number in $[0, 1]$, which is equivalent to a random initialization of a single universe. Subsequently, a new universe inflation rate is detected and compared with the original best universe inflation rate before the jump perturbance. If it is better, then the new position is screened as the best universe; otherwise, the original best universe will be kept.

This design of adding multiple random jump enables the improved algorithm to be able to directly jump out the local optimum situation, so that other universes can also continue to explore and exploit away from the local optimum. Hence, the improved algorithm can avoid falling into a locally optimal situation.

3.5. **The process of RISEMVO algorithm.** The overall process of the RISEMVO algorithm based on the above modifications is as follows:

1) Defining parameters, including the number of universes Num ($then\ i = 1$ to Num), the problem dimension Dim ($then\ j = 1$ to Dim), TDR, TDR_2, the number of total algorithm iterations $t_{\max}$, and the problem boundaries $[lb, ub]$;

2) Initializing each universe to a position within the problem boundary $[lb, ub]$;

3) Starting the optimization iteration, according to the corresponding fitness function (e.g., test function in Section 4) value, calculating the inflation rate of each universe, i.e., adapting to the problem and sorting it, and selecting the universe of a biggest inflation rate as best universe;

4) Selecting a black hole (the current universe inflation rate is compared with a random number in $[0, 1]$, if smaller, select the universe), using the inflation rates as input factor for roulette to select a universe as a white hole, and passing the white hole coordinates to the black hole;

5) Selecting the universe with half probability for wormhole crossing; for the universe crossed by the wormhole to the best universe for nearby exploration and exploitation, updating the location of the universe according to Equation (6);

6) For the universes which are not selected for wormhole crossing, making them orbiting the best universe according to Equation (7);

7) Applying jump perturbance on the best universe to avoid the local optimum situation according to Section 3.4;

8) Determining if the number of total iterations is reached. If not, skip to 3) and continue. If the maximum number of iterations is reached, output the best universe position as the optimal solution and end the current loop.

The pseudo-code of RISEMVO algorithm is shown in Algorithm 1.

4. **Results of CEC Benchmark Experiment.** In this part, a series of experiments was conducted to evaluate the performance of the proposed RISEMVO algorithm. Although there are many other test sets, in order to verify the feasibility of the improvements under the same indicators, we use the test set used by the original authors of MVO algorithm. The numerical efficiency of the proposed RISEMVO algorithm is benchmarked on 29 test functions [11,12], which is compared with these of 5 other heuristic algorithms, including standard MVO algorithm, whale optimization algorithm (WOA) [19], factored gray wolf optimization algorithm (FGWO) [20,21], salp swarm algorithm (SSA) [22,23] and a classical heuristic algorithm, particle swarm optimization (PSO) [24]. And fitness function is the test function here.

29 test functions can be divided into four groups: unimodal, multimodal, fixed-dimension multimodal, and composite functions. Different types of functions benchmark the performance of the algorithms from different perspectives. These functions are listed in Table 1, Table 3, Table 4 and Table 7, and are numbered successively from F_1 to F_{29}. In F_1-F_{23}, 30 search agents were employed by each algorithm to conduct optimization over 100 iterations, and the algorithms were tested repeatedly for more than 10 times. To further investigate the performance exploitation and exploration abilities, the convergence speed, and the local jump ability, in F_{24}-F_{29}, the maximum number of iterations was reduced to 50 times, and the algorithms would be also tested repeatedly for more than 10 times. For all test functions, the number of universes Num in optimization search was set to be 30.

The average value (Ave) of the optimal parameters and the standard deviation (Std) of the fitness function (test function here) values are used to evaluate the performance of the algorithms. These two parameters were adopted according to the fitness, and the resultant meanings were also different. In the CEC Benchmark test used in this

Algorithm 1. Pseudo-code of RISEMVO algorithm

Create Num random universes x^i $(i = 1, 2, \ldots, Num)$
Initialize WEP, TDR, TDR_2 *and Best_universe*
SU = *Sorted universes*
NI = *Normalize the inflation rate (e.g., finesses of test function in Section 4) of the universes*
while *The maximum number* $t_{\max}$ *of iterations is not reached*
  Evaluate the inflation rate of all universes
  for *each universe indexed by i*
    Update TDR and TDR_2
    Black_hole_index = i;
    for *each object indexed by j*
      $r_1 = random([0, 1])$;
      if $r_1 < NI(x^i)$
        White_hole_index = RouletteWheelSelection(-NI);
        $x_j^{Black\_hole\_index} = SU(White\_hole\_index, j)$;
      end if
      $r_2 = random([0, 1])$;
      if $r_2 < WEP$
        Update the location of the universe according to Equation (6)
      else if $r_2 > WEP$
        Update the location of the universe according to Equation (7)
      end if
    end for
  end for
      Applying jump perturbance on the best universe according to Section 3.4
end while

Table 1. Unimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$F_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$	30	$[-100, 100]$	0
$F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
$F_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	$[-100, 100]$	0
$F_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	$[-1.28, 1.28]$	0

work, smaller *Ave* and *Std* represent for better optimizing results. In addition, in order to investigate the similarity of the exploration process among the algorithms, the *p-value* proposed by Wilcoxon Ranksum [25] is used. A smaller *p-value* indicates a larger difference in the exploration process between the strategy of current algorithm and these of other algorithms. By definition, when *p-value* is less than 0.05, the difference is considered to be significant.

TABLE 2. Test results of the algorithms on unimodal benchmark functions

F	Type	RISEMVO	MVO	FGWO	WOA	SSA	PSO
F_1	*Ave*	**9.9022e-261**	35.9324	1.4569e-08	7.8608e-12	1127.2019	5.5524
	Std	**0**	13.201	1.2318e-08	1.7959e-11	488.9175	2.7165
	p-value	**N/A**	4.111e-12	4.111e-12	4.111e-12	4.111e-12	4.111e-12
F_2	*Ave*	**1.0598e-139**	40.6241	5.5608e-06	9.9303e-09	17.4204	12.7292
	Std	**5.8048e-139**	38.4985	2.3173e-06	2.249e-08	4.0798	7.0426
	p-value	**N/A**	3.1602e-12	3.1602e-12	3.1602e-12	3.1602e-12	3.1602e-12
F_3	*Ave*	**7.0348e-252**	6500.6941	36.3872	99088.4216	6887.5418	1389.2315
	Std	**0**	1351.1032	76.4156	21955.9083	4814.6297	419.7865
	p-value	**N/A**	1.102e-11	1.102e-11	1.102e-11	1.102e-11	1.102e-11
F_4	*Ave*	**2.4097e-128**	23.8953	0.036002	62.8841	20.8953	5.5333
	Std	**1.3184e-127**	7.211	0.025973	27.2083	3.6238	1.3127
	p-value	**N/A**	2.2623e-11	2.2623e-11	2.2623e-11	2.2623e-11	2.2623e-11
F_5	*Ave*	28.6273	3889.6155	**27.5104**	28.7925	187627.7897	1901.2013
	Std	**0.17048**	4684.289	0.4965	0.45943	151577.648	677.143
	p-value	**N/A**	3.0199e-11	5.5329e-08	1.0188e-05	3.0199e-11	3.0199e-11
F_6	*Ave*	**0.82865**	35.2877	1.9127	1.9101	1015.8712	5.2375
	Std	**0.29591**	7.1788	0.36006	0.50409	339.1952	2.2999
	p-value	**N/A**	3.0199e-11	6.6955e-11	1.6132e-10	3.0199e-11	3.3384e-11
F_7	*Ave*	**0.00035527**	0.14631	0.011336	0.016339	0.61067	5.322
	Std	**0.00033844**	0.053715	0.0057112	0.014411	0.21426	5.1024
	p-value	**N/A**	3.0199e-11	3.0199e-11	1.7769e-10	3.0199e-11	3.0199e-11

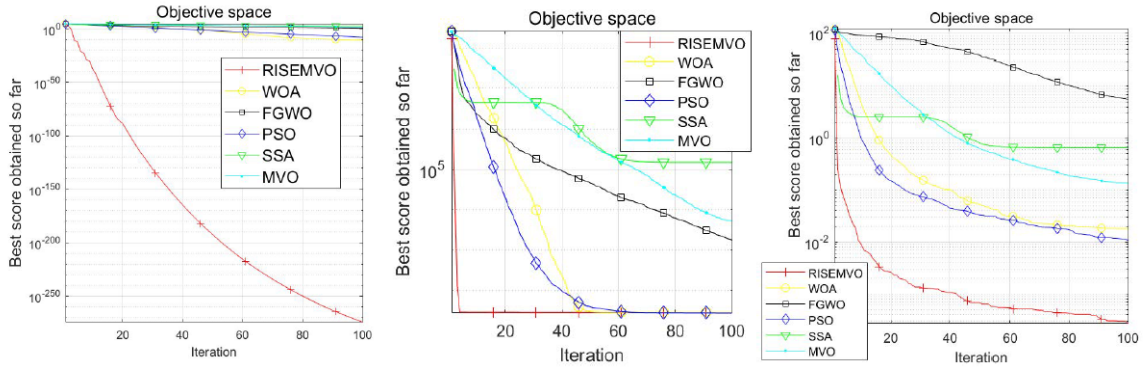

FIGURE 1. Convergence curves of the algorithms tested on unimodal benchmark functions (F_1, F_5, F_7)

4.1. **Evaluation of exploitation capability.** Unimodal functions F_1-F_7 have only one global optimum. As shown in Table 1, these functions allow to evaluate the exploitation capability of one algorithm. Here, Dim stands for the dimension of the functions, Range is the boundary of the function's search space, and $f_{\min}$ is the optimum of test functions. Results are shown in Table 2. In order to observe the convergence behavior of RISEMVO algorithm, the convergence curves of the algorithms tested on F_1, F_5, F_7 are shown in Figure 1.

It can be found that RISEMVO algorithm presents the smallest *Ave* or *Std* value in most test functions, which proves that RISEMVO algorithm performs best in almost all the test functions. Although the lowest *Ave* value is not obtained by RISEMVO algorithm in the function F_5, RISEMVO algorithm still exhibits good performance. Hence, RISEMVO algorithm shows notable improvement in exploitation ability compared with the standard MVO algorithm. Meanwhile, *p-value* is always kept at a very small number,

revealing the unique feature of RISEMVO algorithm compared with other algorithms. The improvement of *TDR* makes the RISEMVO algorithm keeping a slower iteration trend in the later stage of the iteration for local exploiting convergence; at the same time, the emergence of TDR_2 also promotes the in-depth exploiting of the optimal solution for the later iteration. Hence, RISEMVO preforms excellent in exploitation capability.

TABLE 3. Multimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]$	-418.9829×5
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	$[-5.12, 5.12]$	0
$F_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)$ $-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	$[-32, 32]$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$	0
$F_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right]\right.$ $\left. + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$F_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right]\right.$ $\left. + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	$[-50, 50]$	0

TABLE 4. Fixed-dimension multimodal benchmark functions

Function	Dim	Range	$f_{\min}$
$F_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	$[-65, 65]$	0
$F_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	$[-5, 5]$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316
$F_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	$[-5, 5]$	0.398
$F_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2\right.\right.$ $\left.\left. + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2\right.\right.$ $\left.\left. + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	$[-2, 2]$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$	3	$[1, 3]$	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$	6	$[0, 1]$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	$[0, 10]$	-10.5363

4.2. **Evaluation of exploration capability.** The second and third types of the benchmark functions have many local optima whose number increases exponentially with the number of variables. As shown in Table 3 and Table 4, the multimodal functions F_8-F_{23} are designed for evaluating the exploration capabilities of the algorithms. The search may fall into local optimums if the algorithm deals with multimodal functions. Even if the algorithm owns strong exploitation ability, lacking of the exploration ability would make all efforts in vain.

As can be seen from the results listed in Table 5 and Table 6, RISEMVO algorithm presents the smallest *Ave* value in most test functions; meanwhile, the *p-value* is small for most test functions. These indicate that RISEMVO algorithm has a high level of exploration ability and has obtained a unique way of exploration. And in order to observe the convergence behavior of RISEMVO algorithm, convergence curves of the algorithms tested on F_{10}, F_{12}, F_{15} are shown in Figure 2. The performance of RISEMVO algorithm correlates with the modification of *TDR*. The increase in the initial value of *TDR* and the appearance of TDR_2 increase the early exploration of the RISEMVO algorithm when the number of iterations is small. At the same time, increasing the revolution of the universe also increases the possibility that the RISEMVO algorithm explores a better solution. Thence RISEMVO algorithm behaves better in exploration compared with the original MVO algorithm and other algorithms.

TABLE 5. Test results of the algorithms on multimodal benchmark functions

F	Type	RISEMVO	MVO	FGWO	WOA	SSA	PSO
F_8	*Ave*	**−12477.5489**	−7357.3528	−3121.7807	−9368.7528	−6124.1248	−3170.4613
	Std	**34.5976**	574.0314	340.9699	1471.225	693.5423	397.4943
	p-value	**N/A**	3.0199e-11	3.0199e-11	5.0723e-10	3.0199e-11	3.0199e-11
F_9	*Ave*	**0**	157.1927	39.6211	4.116	90.5208	221.5775
	Std	**0**	32.6734	35.4434	20.149	17.1791	36.0617
	p-value	**N/A**	1.2088e-12	1.2088e-12	1.2088e-12	1.2088e-12	1.2088e-12
F_{10}	*Ave*	**8.8818e-16**	3.7377	2.0928e-05	5.154e-07	9.3333	2.957
	Std	**0**	0.46217	1.0858e-05	9.473e-07	1.0377	0.38913
	p-value	**N/A**	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12
F_{11}	*Ave*	**0**	1.324	0.029528	0.024116	11.4721	19.6807
	Std	**0**	0.08172	0.057157	0.11829	3.2236	5.2257
	p-value	**N/A**	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12
F_{12}	*Ave*	**0.045266**	8.2459	0.15443	0.16028	1399.825	0.81828
	Std	**0.015822**	4.4539	0.046363	0.080558	7152.7747	0.94073
	p-value	**N/A**	3.0199e-11	3.4742e-10	6.5183e-09	3.0199e-11	3.0199e-11
F_{13}	*Ave*	**0.49634**	16.1757	1.5019	1.24	17178.4994	1.9926
	Std	**0.14277**	17.7872	0.22923	0.43019	33150.6531	1.0037
	p-value	**N/A**	3.0199e-11	3.0199e-11	6.0658e-11	3.0199e-11	5.4941e-11

4.3. **Evaluation of avoiding local optimums.** Composite benchmark test functions shown in Table 7, are the combination of several basic benchmark functions with the same domain of definition, are quite complex and have many local optimal solutions. Only when the exploitation and exploration abilities are strictly balanced, the algorithm can jump out of the local optimal solutions. The test results of the algorithms are shown in Table 8. In order to observe the convergence behavior of RISEMVO algorithm, the convergence curves of the algorithms tested on F_{24}, F_{29} are shown in Figure 3.

According to the results, RISEMVO algorithm still exhibits the best performance among all algorithms for functions F_{24} and F_{29}, and presents good performance for the

TABLE 6. Test results of the algorithms on fixed-dimension multimodal benchmark functions

F	Type	RISEMVO	MVO	FGWO	WOA	SSA	PSO
F_{14}	*Ave*	**0.998**	2.635	5.4652	5.5588	3.8792	3.3926
	Std	**4.7091e-06**	2.9815	4.5913	4.7332	3.4493	3.0732
	p-value	**N/A**	0.4553	3.3384e-11	6.121e-10	0.00039718	0.0079494
F_{15}	*Ave*	**0.00066354**	0.0033012	0.0027659	0.00099705	0.0052946	0.0037565
	Std	**0.0003206**	0.0060562	0.0059768	0.00062242	0.0077555	0.0066386
	p-value	**N/A**	2.9215e-09	1.8608e-06	0.00031821	5.0723e-10	4.6159e-10
F_{16}	*Ave*	**−1.0315**	−1.0316	−1.0316	−1.0316	−1.0316	−1.0316
	Std	0.00013008	1.0057e-05	1.9404e-05	8.1482e-07	2.2445e-13	**2.9514e-13**
	p-value	**N/A**	7.5991e-07	0.00037704	3.6897e-11	3.0104e-11	2.9991e-11
F_{17}	*Ave*	0.39795	**0.39789**	0.39866	0.39887	**0.39789**	**0.39789**
	Std	9.7069e-05	3.3781e-06	0.000607	0.0013599	7.8405e-14	**6.6817e-15**
	p-value	**N/A**	1.0277e-06	2.6695e-09	1.1077e-06	3.0142e-11	1.6051e-11
F_{18}	*Ave*	3.0018	3.0001	3.0004	5.7326	**3**	**3**
	Std	0.0031974	9.5606e-05	0.00053121	8.3325	2.5759e-12	**8.2278e-13**
	p-value	**N/A**	1.3594e-07	0.0072884	0.25188	3.0199e-11	3.018e-11
F_{19}	*Ave*	−3.8112	−3.8627	−3.8595	−3.8306	−3.8594	**−3.8628**
	Std	0.19613	4.9865e-05	0.0033059	0.045266	0.0057231	**2.4592e-12**
	p-value	**N/A**	0.55923	9.2603e-09	1.3111e-08	0.00065486	3.0199e-11
F_{20}	*Ave*	**−3.2782**	−3.2629	−3.2406	−3.1251	−3.1883	−3.2687
	Std	**0.05833**	0.069848	0.061464	0.12801	0.1107	0.080166
	p-value	**N/A**	0.011228	5.265e-05	5.1857e-07	0.39527	0.33285
F_{21}	*Ave*	−5.8184	−5.552	−5.4333	**−7.4022**	−7.0705	−6.0672
	Std	**2.0125**	3.2191	2.1275	2.4845	3.6324	3.4991
	p-value	**N/A**	0.83026	0.18577	0.8883	0.093341	0.5106
F_{22}	*Ave*	−5.8634	−7.1757	−5.7218	−5.7251	**−8.2661**	−7.2712
	Std	2.2144	3.5827	**1.433**	2.4521	3.3652	3.4992
	p-value	**N/A**	0.21156	0.8883	0.00062027	0.00085641	0.032651
F_{23}	*Ave*	−5.3875	−7.5496	−6.0112	−6.8004	−6.2744	**−7.7319**
	Std	**1.4561**	3.7665	1.8721	2.8946	3.8087	3.4961
	p-value	**N/A**	0.13345	0.19579	0.0027548	0.14945	0.029205

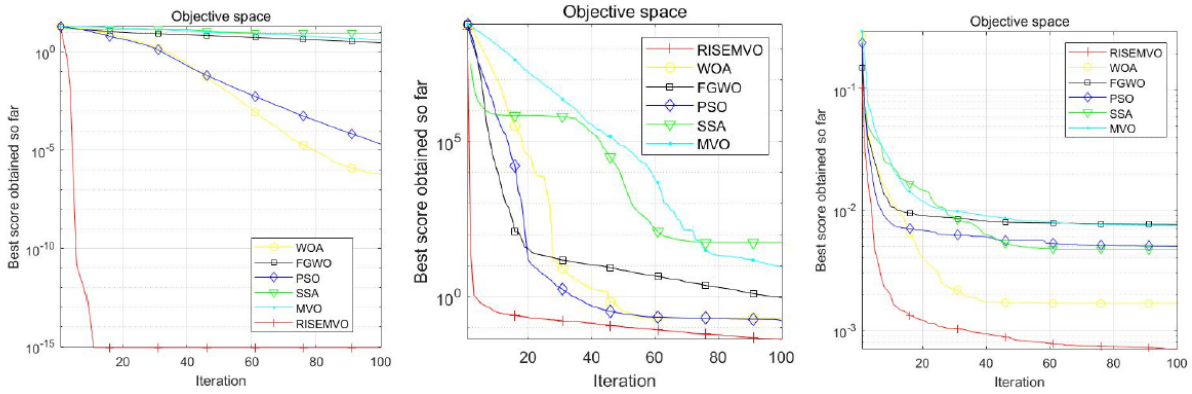

FIGURE 2. Convergence curves of the algorithms tested on multimodal functions (F_{10}, F_{12}, F_{15})

TABLE 7. Composite benchmark functions

Function	Dim	Range	$f_{\min}$
$F_{24}(CF1)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10} =$ *Sphere Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3 \ldots, \lambda_{10}] = [5/100, 5/100, 5/100, \ldots, 5/100]$			
$F_{25}(CF2)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10} =$ *Griewank's Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [5/100, 5/100, 5/100, \ldots, 5/100]$			
$F_{26}(CF3)$	30	$[-5,5]$	0
$f_1, f_2, f_3, \ldots, f_{10} =$ *Griewank's Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [1, 1, 1, \ldots, 1]$			
$F_{27}(CF4)$	30	$[-5,5]$	0
$f_1, f_2 =$ *Ackley's Function*,			
$f_3, f_4 =$ *Rastrigin's Function*,			
$f_5, f_6 =$ *Weierstrass Function*,			
$f_7, f_8 =$ *Griewank's Function*,			
$f_9, f_{10} =$ *Sphere Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [5/32, 5/32, 1, 1, 5/0.5, 5/0.5,$			
$5/100, 5/100, 5/100, 5/100]$			
$F_{28}(CF5)$	30	$[-5,5]$	0
$f_1, f_2 =$ *Rastrigin's Function*,			
$f_3, f_4 =$ *Weierstrass Function*,			
$f_5, f_6 =$ *Griewank's Function*,			
$f_7, f_8 =$ *Ackley's Function*,			
$f_9, f_{10} =$ *Sphere Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [1, 1, 1, \ldots, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [1/5, 1/5, 5/0.5, 5/0.5, 5/100,$			
$5/100, 5/32, 5/32, 5/100, 5/100]$			
$F_{29}(CF6)$	30	$[-5,5]$	0
$f_1, f_2 =$ *Rastrigin's Function*,			
$f_3, f_4 =$ *Weierstrass Function*,			
$f_5, f_6 =$ *Griewank's Function*,			
$f_7, f_8 =$ *Ackley's Function*,			
$f_9, f_{10} =$ *Sphere Function*,			
$[\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_{10}] = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$,			
$[\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_{10}] = [0.1 * 1/5, 0.2 * 1/5, 0.3 * 5/0.5, 0.4 * 5/0.5,$			
$0.5 * 5/100, 0.6 * 5/100, 0.7 * 5/32, 0.8 * 5/32, 0.9 * 5/100, 1 * 5/100]$			

remaining test functions. This means that the RISEMVO algorithm has a good ability to jump out of the local optimal solutions. This good performance benefits from the *WEP* settings. *WEP* has been maintained at 0.5 instead of gradually increasing to 1, which avoids the possibility of falling into the local optimal problem in the later iteration. And even if it falls into a local optimum, it can get out of the dilemma through the setting out of the local optimal strategy in Section 3.4.

TABLE 8. Test results of the algorithms on composite benchmark functions

F	Type	RISEMVO	MVO	FGWO	WOA	SSA	PSO
F_{24}	*Ave*	**52.7587**	99.9963	198.9244	545.9576	442.477	311.55
	Std	89.6551	129.5506	**82.6089**	185.5975	245.6454	155.1197
	p-value	**N/A**	3.8349e-06	1.7294e-07	4.9752e-11	6.121e-10	2.2273e-09
F_{25}	*Ave*	618.6507	**338.1578**	411.4573	635.109	699.448	404.7498
	Std	342.0678	135.552	167.3933	**133.8132**	254.3437	145.3798
	p-value	**N/A**	0.011249	0.0060653	0.26522	0.38165	0.052786
F_{26}	*Ave*	880.7873	**614.7434**	764.1994	1077.2073	1011.384	712.0838
	Std	105.2321	167.3203	110.2262	155.3277	264.4073	**101.6427**
	p-value	**N/A**	7.4716e-10	1.3341e-08	7.4716e-10	0.15722	3.3593e-11
F_{27}	*Ave*	874.3306	**776.6633**	875.1667	926.4215	1023.347	814.2568
	Std	**0**	147.6834	57.3396	76.0258	61.8027	139.7142
	p-value	**N/A**	0.082015	0.027312	2.3657e-12	2.3657e-12	0.00055065
F_{28}	*Ave*	660.8668	**145.3499**	308.3523	587.3162	703.4362	371.14
	Std	377.8436	186.3315	250.4706	221.2849	401.6797	**175.2079**
	p-value	**N/A**	0.00015348	0.018034	0.48585	0.0071395	0.0027035
F_{29}	*Ave*	**900**	902.7868	900.0008	900.0001	943.3333	964.8246
	Std	**0**	18.5206	0.00047184	4.8301e-05	9.3035	15.366
	p-value	**N/A**	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12

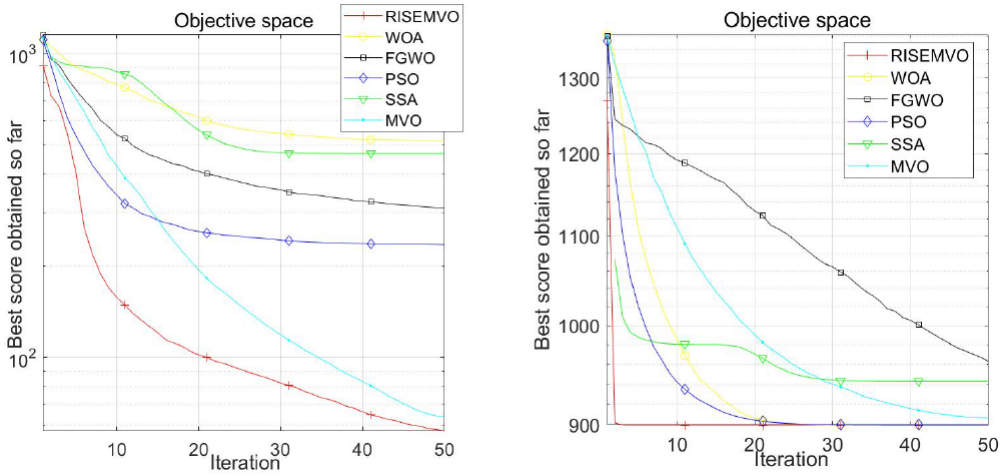

FIGURE 3. Convergence curves of the algorithms tested on composite benchmark functions (F_{24}, F_{29})

It can also be seen from the convergence curve shown in Figures 1-3 that the RISEMVO algorithm works well in these test functions, and can maintain the convergence rate among the 29 test functions in the first few of the 6 test functions. Compared with the original MVO algorithm, we can see RISEMVO algorithm is superior to the original algorithm in the three types of test functions.

Meanwhile, we ranked the average performance of all algorithms by Friedman test [25]. The ranking results are shown in Table 9, and the smaller the ranking value, the better the algorithm performance. According to these statistical results, we can see that by improving the original MVO algorithm in 4 steps, RISEMVO algorithm can achieve better performance than other comparison algorithms in test functions.

TABLE 9. Average ranking values using Friedman test in different types of test functions

Algorithm	Unimodal Function	Multimodal Function	Fixed-Multimodal Function	Composite Function	Total Ranking
RISEMVO	1.14	1	3.3	3	2.2413
MVO	4.86	4.1667	2.7	1.6676	3.3103
FGWO	2.29	3.3333	4.3	2.6667	3.2759
WOA	3.43	2.1667	4.3	4.6667	3.7241
SSA	5.29	5.3333	3	5.3333	4.5172
PSO	3.57	4.3333	2.1	3.5	3.2069

Overall, considering all the test results of CEC benchmark functions, it can be concluded that RISEMVO algorithm has superior exploitation ability, exploration ability, and the ability to jump out of local optimality compared with the comparison algorithms.

5. **Application on Web Device Resource Allocation Problem.** Due to the limited resources available on the media presentation device, inappropriate resource allocation scheme may cause playback interruption or unnecessary waste of resources in the media presentation process. In addition, as the tasks during the media presentation have different resource priorities and successive dependencies, they bring inconvenience to resource allocation. An example (Google Chrome browser) of Web browser architecture and process invocation are shown in Figure 4. Here, HTML, SVG, CSS, DOM, UI, GPU and 2D/3D represent for Hyper Text Markup Language, Scalable Vector Graphics, Cascading Style Sheets, Document Object Model, User Interface, Graphics Processing Unit, and 2-Dimension/3-Dimension, respectively.

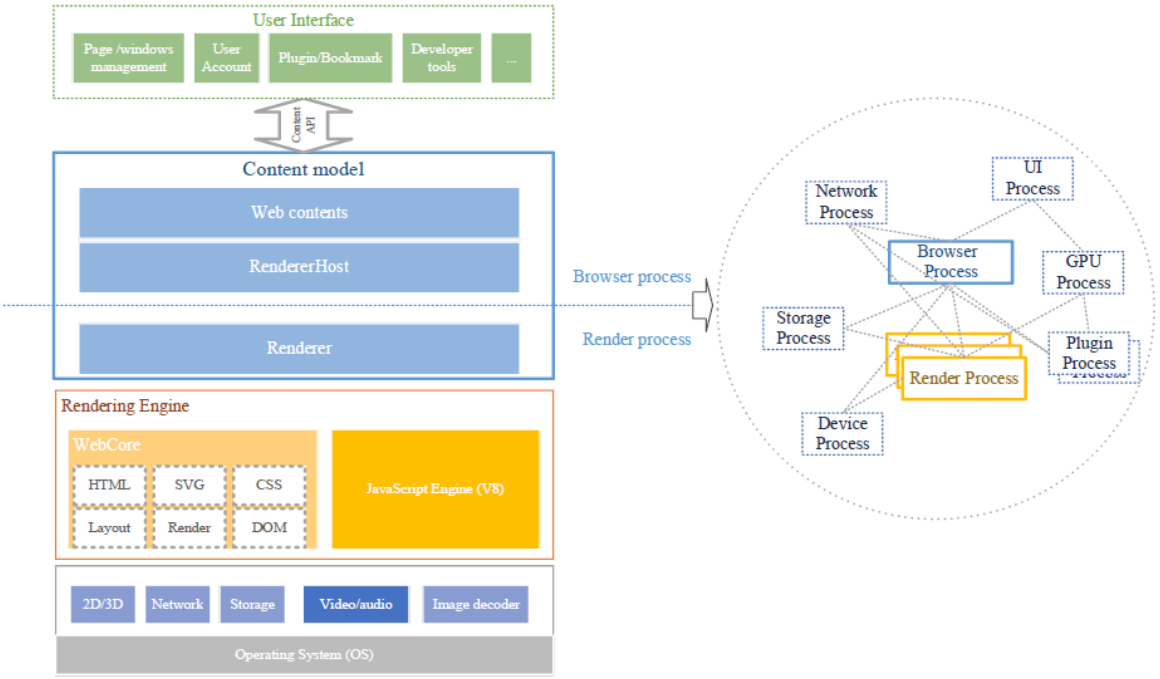

FIGURE 4. Chrome browser architecture and process invocation

5.1. **Multi-resource allocation problem model.** The multi-resource allocation problem is described as follows. The number of total requested resource tasks is M, and the number of current resources is N. For each task T_j $(0 < j \leq M)$, its resource priority weight coefficient is β_j $(0 \leq \beta_j \leq 1)$, which reflects the priority level of the task to obtain resources in the device.

Then available resources are defined as $\{R_1, R_2, \ldots, R_N\}$, and tasks are defined as $\{T_1, T_2, \ldots, T_M\}$. Suppose that the resource R_i acquired by task T_j is R_{ij} $(1 \leq i \leq N, 1 \leq j \leq M)$, and $R_{i1} + R_{i2} + \cdots + R_{iM} = \sum_{j=1}^{M} R_{ij} \leq R_i$, $1 \leq i \leq N$. The resource demand of R_i by T_j has a minimum value of $R_{ij}^{\min}$ and a maximum value of $R_{ij}^{\max}$. $R_{ij}^{\min}$ must satisfy the lowest Quality of Service (QoS). We can improve QoS by adding the resource allocation R_i for $R_i < R_{ij}^{\max}$, until the QoS cannot be further improved. For task T_j, the abstract modeling setting of the context dependency of the task is $\sum_{i=1}^{N} \delta_{i1} R_{i1} \geq \sum_{i=1}^{N} \delta_{i2} R_{i2} \geq \cdots \geq \sum_{i=1}^{N} \delta_{iM} R_{iM}$, where δ_{ij} $(1 \leq i \leq N,\ 1 \leq j \leq M,\ 0 \leq \delta_{ij} \leq 1)$ is the resource conversion factor of the current task output. QoS uses utility to measure, and the utility function of task T_j is $q_j = \beta_j f_j\ (R_{1j}, R_{2j}, \ldots, R_{Nj})$, and $q_{j\min} = q_j\left(R_{1j}^{\min}, R_{2j}^{\min}, \ldots, R_{Nj}^{\min}\right)$ and $q_{j\max} = q_j\left(R_{1j}^{\max}, R_{2j}^{\max}, \ldots, R_{Nj}^{\max}\right)$. The utility function tools used for different tasks may be different, which brings difficulties to judge the overall performance of the system. To reduce differentiation and obtain better optimization results, the utility function is standardized for various tasks. The standardized utility function of each task of the resource allocation model is as follows:

$$Q_j\left(R_{1j}, \ldots, R_{Nj}\right) = \begin{cases} 0, & R_{ij} = R_{ij}^{\min} \\ FUN(q_j), & R_{ij}^{\min} < R_{ij} < R_{ij}^{\max} \\ 1, & R_{ij} = R_{ij}^{\max} \end{cases} \tag{8}$$

where $1 \leq i \leq N$, $1 \leq j \leq M$, and $0 \leq Q_j \leq 1$. $FUN(q_j)$ meets two marginal conditions: 1) the minimum resource acquisition is 0 while the maximum resource acquisition is 1; 2) $FUN(q_j)$ is positively correlated with the size of resources.

5.2. **System utility maximization model.** To construct a model for maximized utilizing the system with N resources upon M tasks, we need to search for an $N \times M$ resource allocation matrix U_{ij} to maximize the utility of all resource. The system utility maximization model can be formulated as follows:

$$\max \sum_{j=1}^{M} Q_j\left(R_{1j}, R_{2j}, \ldots, R_{Nj}\right) \tag{9}$$

where Q_j satisfies Equation (8), and R_{ij} satisfies

$$\sum_{j=1}^{M} R_{ij} \leq R_i, \quad 1 \leq i \leq N \tag{10}$$

$$R_{ij}^{\min} \leq R_{ij} \leq R_{ij}^{\max}, \quad 1 \leq i \leq N \tag{11}$$

6. **Performance of RISEMVO Algorithm on Web Device Resource Allocation Problem.** The RISEMVO algorithm is used to solve the multi-resource allocation problem model to maximize the system utility.

Different from the optimal minimum value in the test set, the optimal solution sought in this section is the solution that maximizes the system's utility. That is, the larger the *Ave* value, the better the algorithm performance. Other parameters remain unchanged in specific experiments. According to the number of the total requested resources and the number of the total tasks in the actual Web media rendering process, we set the number

of the total resource types in the problem model is $N = 4$ and the number of total tasks is $M = 10$. Referring to the settings in [17], we choose four normalized utility functions $FUN(q_j)$. The utility function q_j contains the priority information of the task to the resource acquisition. In this experiment, the priority of each task is set to be different, which has ten levels varying from 0 to 1. The normalized utility functions are as follows:

1) $FUN(q_j) = \frac{\sum_{i=1}^{4} R_{ij} - \sum_{i=1}^{4} R_{ij}^{\min}}{\sum_{i=1}^{4} R_{ij}^{\max} - \sum_{i=1}^{4} R_{ij}^{\min}}$;

2) $FUN(q_j) = \frac{\prod_{i=1}^{4} R_{ij} - \prod_{i=1}^{4} R_{ij}^{\min}}{\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}^{\min}}$;

3) $FUN(q_j) = \sin\left(\frac{\pi}{2} \times \frac{\sum_{i=1}^{4} R_{ij} - \sum_{i=1}^{4} R_{ij}^{\min}}{\sum_{i=1}^{4} R_{ij}^{\max} - \sum_{i=1}^{4} R_{ij}^{\min}}\right)$;

4) $FUN(q_j) = 1 - \sin\left(\frac{\pi}{2} \times \frac{\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}}{\prod_{i=1}^{4} R_{ij}^{\max} - \prod_{i=1}^{4} R_{ij}^{\min}}\right)$.

The total available resources of each type are 1, and the resource requirements of 10 tasks are shown in Table 10 [17]. Each algorithm employs 30 universes ($Num = 30$) and runs 100 iterations. In order to reduce the accidental factors, each algorithm is tested repeatedly. The results are shown in Table 11 and Figure 5.

TABLE 10. Task type and resource requirement on Web device resource allocation problem

Task	Minimum resource requirement	Maximum resource requirement	Task type
1	(0.01,0.01,0.01,0.01)	(0.3,0.25,0.22,0.25)	1
2	(0.02,0.03,0.01,0.03)	(0.32,0.3,0.2,0.2)	1
3	(0.03,0.02,0.03,0.01)	(0.28,0.22,0.38,0.19)	1
4	(0.01,0.02,0.01,0.02)	(0.3,0.4,0.5,0.35)	2
5	(0.01,0.01,0.02,0.03)	(0.2,0.3,0.3,0.3)	2
6	(0.02,0.03,0.01,0.02)	(0.38,0.3,0.32,0.29)	2
7	(0.01,0.01,0.02,0.02)	(0.19,0.18,0.23,0.26)	3
8	(0.01,0.01,0.01,0.01)	(0.32,0.28,0.19,0.18)	3
9	(0.02,0.01,0.02,0.01)	(0.35,0.4,0.2,0.3)	4
10	(0.02,0.03,0.03,0.01)	(0.25,0.3,0.35,0.2)	4

TABLE 11. Comparison of the performances of different algorithms on Web device resource allocation problem

Type	RISEMVO	MVO	FGWO	WOA	SSA	PSO
Ave	**3.9986**	3.4578	3.8782	3.1206	3.4476	3.8037
Std	**0.0007809**	0.12802	0.033008	0.12574	0.18265	0.1278
p-value	**N/A**	7.0661e-18	7.0661e-18	7.0661e-18	7.0661e-18	2.2917e-15

The results of this experiment demonstrate that the proposed RISEMVO algorithm outperforms other algorithms in all respects. The optimization condition of RISEMVO algorithm reaches the maximum system utility value, higher than the utility value of other algorithms, that means RISEMVO algorithm provides a good solution in solving multiple resource allocation problem and can get better performance under the same resource constraints. At the same time, the *p-value* is very small. These mean that the RISEMVO algorithm has better performance and is very different from other systems.

Hence, RISEMVO algorithm is an excellent algorithm in processing resource allocation problem, and provides an effective solution to the problem of unsmooth playback caused by resource allocation problems in media presentation process. Though equipment resources

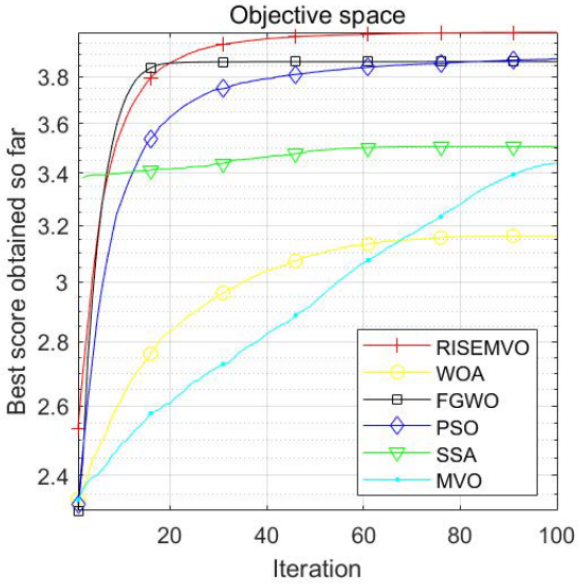

FIGURE 5. Convergence curves for algorithms over resource allocation problem

for media presentation are limited, in the condition of multiple operations running in parallel and consuming various resources, the RISEMVO algorithm enables high system benefits from limited resources. That means that we can avoid the problem of unsmooth media presentation or wasting unnecessary resources caused by inappropriate resource allocation schemes.

7. **Conclusions.** This work proposes an improved multi-verse optimizer algorithm (RISEMVO), in which several strategies are added to optimize the exploration mechanism in search space as well as the exploitation mechanism. Furthermore, the jumping of local optimal strategy is also subjoined. The performance of RISEMVO algorithm was tested on 29 CEC benchmark functions and compared to 5 other algorithms including standard MVO algorithm, and RISEMVO algorithm shows the overall best performance. Furthermore, RISEMVO algorithm also performs best in solving Web device resource allocation problem. These results reveal that RISEMVO algorithm significantly promotes the performance of the standard MVO algorithm and outperforms all the other comparison algorithms.

In future work, we intend to further optimize the RISEMVO algorithm, especially on composite functions. Besides, we also intend to modify the model constraints so as to match more application scenes.

Acknowledgment. The work is partially supported by these projects: Strategic Leadership Project of Chinese Academy of Sciences: SEANET Technology; Standardization Research System Development (Project No. XDC02070100).

REFERENCES

[1] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017-2022*, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.pdf, Accessed in 2019.

[2] *Conviva's State of the Streaming TV Industry Q1 2019*, https://www.conviva.com/research/convivas-state-streaming-tv-industry-q1-2019/, Accessed in 2019.

[3] Z. Wang, H. Deng, X. Zhu and L. Hu, Overview of terminal web runtime and related optimization technology, *Journal of Network New Media*, vol.9, no.01, pp.1-10, 2020.

[4] X. Mi, C. Yang and Z. Chang, Multi-resource management for multi-tier space information networks: A cooperative game, *Proc. of 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, Tangier, Morocco, pp.948-953, 2019.
[5] J. Shi, J. Luo, F. Dong, J. Jin and J. Shen, Fast multi-resource allocation with patterns in large scale cloud data center, *Journal of Computational Science*, vol.26, pp.389-401, doi: 10.1016/j.jocs.2017.05.005, 2018.
[6] F. Fossati, S. Moretti, P. Perny and S. Secci, Multi-resource allocation for network slicing, *IEEE/ACM Trans. Networking*, pp.1-14, doi: 10.1109/TNET.2020.2979667, 2020.
[7] D. Juedes, F. Drews, L. Welch and D. Fleeman, Heuristic resource allocation algorithms for maximizing allowable workload in dynamic, distributed real-time systems, *Proc. of the 18th International Parallel and Distributed Processing Symposium*, 2004.
[8] J. Wang, B. Gong, H. Liu and S. Li, Model and algorithm for heterogeneous scheduling integrated with energy-efficiency awareness, *Transactions of the Institute of Measurement and Control*, vol.38, no.4, pp.452-462, 2016.
[9] S. Mirjalili, S. M. Mirjalili and A. Hatamlou, Multi-verse optimizer: A nature-inspired algorithm for global optimization, *Neural Comput. & Applic.*, vol.27, no.2, pp.495-513, 2016.
[10] S. Mirjalili, P. Jangir, S. Z. Mirjalili, S. Saremi and I. N. Trivedi, Optimization of problems with multiple objectives using the multi-verse optimization algorithm, *Knowledge-Based Systems*, vol.134, pp.50-71, 2017.
[11] X.-S. Yang, Test problems in optimization, in *Engineering Optimization: An Introduction with Metaheuristic Applications*, arXiv preprint arXiv:1008.0549, pp.261-266, 2010.
[12] M. Molga and C. Smutnicki, *Test Functions for Optimization Needs*, 2005.
[13] X. Liu, Application of improved multiverse algorithm to large scale optimization problems, *Journal of Electronics and Information Technology*, vol.41, no.7, pp.1666-1673, 2019.
[14] P. Poullie, T. Bocek and B. Stiller, A survey of the state-of-the-art in fair multi-resource allocations for data centers, *IEEE Transactions on Network and Service Management*, vol.15, no.1, pp.169-183, 2018.
[15] Z. Huan, H. Ni and L. Hu, AAFSA-RA: A multi-resource allocation method based on an advanced artificial fish swarm algorithm, *Journal of Xi'an Jiaotong University*, vol.48, no.10, pp.120-125, 2014.
[16] H. Dai, Y. Yang, J. Yin, H. Jiang and C. Li, Improved greedy strategy for cloud computing resources scheduling, *ICIC Express Letters*, vol.13, no.6, pp.499-504, 2019.
[17] J. D. Barrow, P. C. W. Davies and C. L. Harper, *Science and Ultimate Reality: Quantum Theory, Cosmology, and Complexity*, Cambridge University Press, http://www.loc.gov/catdir/description/cam041/2003055903.html, 2004.
[18] G. I. Sayed, A. Darwish and A. E. Hassanien, Quantum multiverse optimization algorithm for optimization problems, *Neural Comput. & Applic.*, vol.31, no.7, pp.2763-2780, doi: 10.1007/s00521-017-3228-9, 2019.
[19] S. Mirjalili and A. Lewis, The whale optimization algorithm, *Advances in Engineering Software*, vol.95, pp.51-67, 2016.
[20] W. Xiao, H. Deng, Y. Sheng and L. Hu, Factored grey wolf optimizer with application to resource-constrained project to scheduling, *International Journal of Innovative Computing, Information and Control*, vol.14, no.3, pp.881-897, 2018.
[21] S. Mirjalili, S. M. Mirjalili and A. Lewis, Grey wolf optimizer, *Advances in Engineering Software*, vol.69, pp.46-61, 2014.
[22] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris and S. M. Mirjalili, Salp swarm algorithm: A bio-inspired optimizer for engineering design problems, *Advances in Engineering Software*, vol.114, pp.163-191, 2017.
[23] B. Ma, H. Ni, X. Zhu and Z. Wang, A transformed salp swarm algorithm on container deployment problem, *International Journal of Innovative Computing, Information and Control*, vol.16, no.1, pp.283-299, 2020.
[24] J. Kennedy and R. Eberhart, Particle swarm optimization, *International Conference on Neural Networks (ICNN'95)*, Perth, WA, Australia, pp.1942-1948, 1995.
[25] J. Derrac, S. García, D. Molina and F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm and Evolutionary Computation*, vol.1, no.1, pp.3-18, 2011.