Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Improved dynamic grey wolf optimizer*

Xiaoqing ZHANG†‡[1,2], Yuye ZHANG[1], Zhengfeng MING[2]

[1]*School of Physics and Electronic Engineering, Xianyang Normal University, Xianyang 712000, China*

[2]*School of Mechano-Electronic Engineering, Xidian University, Xi'an 710071, China*

†E-mail: 249140543@qq.com

Received Apr. 24, 2020; Revision accepted July 9, 2020; Crosschecked May 17, 2021

Abstract: In the standard grey wolf optimizer (GWO), the search wolf must wait to update its current position until the comparison between the other search wolves and the three leader wolves is completed. During this waiting period, the standard GWO is seen as the static GWO. To get rid of this waiting period, two dynamic GWO algorithms are proposed: the first dynamic grey wolf optimizer (DGWO1) and the second dynamic grey wolf optimizer (DGWO2). In the dynamic GWO algorithms, the current search wolf does not need to wait for the comparisons between all other search wolves and the leading wolves, and its position can be updated after completing the comparison between itself or the previous search wolf and the leading wolves. The position of the search wolf is promptly updated in the dynamic GWO algorithms, which increases the iterative convergence rate. Based on the structure of the dynamic GWOs, the performance of the other improved GWOs is examined, verifying that for the same improved algorithm, the one based on dynamic GWO has better performance than that based on static GWO in most instances.

Key words: Swarm intelligence; Grey wolf optimizer; Dynamic grey wolf optimizer; Optimization experiment
https://doi.org/10.1631/FITEE.2000191 **CLC number:** TP301

1 Introduction

The grey wolf optimizer (GWO) is a swarm intelligence algorithm that imitates the hunting behavior of grey wolves (Mirjalili et al., 2014). In the GWO iteration process, the search is done mainly according to the positions of these leading wolves, wolves α, β, and δ (Gupta and Deep, 2019a). In GWO, the search wolves would spread out to find prey, and would work together to attack prey. In the optimization process, the search information of the previous iteration is retained. Wolf α is applied to save the best value in memory until the last iteration, and the ω wolves search the optimal value in parallel to accelerate convergence. GWO has a simple structure that is easy to understand and has few initialization parameters, which means it can be quickly applied in many fields (Zhang S and Zhou, 2015; Mirjalili et al., 2016).

Although GWO has many advantages, there are also some shortcomings. For example, because the continuous decimal is applied to express the information, it would be not convenient for use in some applications with discrete or binary expressions. Thus, the GWO needs to be improved with respect to its information representation. Some studies have been done. To conveniently describe the feature selection problem, two binary GWO algorithms have been proposed: binary grey wolf optimizer algorithm 1 (bGWO1) and binary grey wolf optimizer algorithm 2 (bGWO2) (Emary et al., 2016). Aiming at the scheduling problem in the welding process, a multiobjective discrete GWO (MODGWO) has been proposed based on establishing its programming model (Lu et al., 2016).

In GWO, the sole parameter, parameter $\boldsymbol{a}$, is larger in earlier iterations to enhance the exploration

‡ Corresponding author

* Project supported by the Scientific Research Plan Projects of Shaanxi Education Department (No. 20JK0972), the Natural Science Basic Research Project of Shaanxi Province (No. 2021JM-517), and the Educational and Teaching Reform Research Project of Xianyang Normal University (No. 2017Y004)

ORCID: Xiaoqing ZHANG, https://orcid.org/0000-0001-8939-2570

ability, and decreases in later iterations to strengthen the exploitation capacity, which makes the GWO structure simple. Although the exploration and the exploitation performances have been balanced by parameter $\boldsymbol{a}$ in the whole iterative process, the performance of the GWO is limited because its exploration ability is impaired in the late iterations. Aiming at the issue of parameter $\boldsymbol{a}$, there have been several variants of GWO. The β-chaotic sequence is embedded in parameter $\boldsymbol{a}$ with a normalization mapping method to strengthen the exploration and development ability, and is called β-GWO (Saxena et al., 2019). An improved adaptive GWO (IAGWO) was proposed (Liu et al., 2019), in which parameter $\boldsymbol{a}$ was dynamically adjusted and the inverse parabolic diffusion distribution was adopted to pull wolves back into the feasible region. To increase the search performance, a simple and efficient augmentation was introduced for GWO (AGWO) (Qais et al., 2018), and the behavior of the control parameter $\boldsymbol{a}$ and position updating were modified to increase the possibility of the exploration process over the exploitation process.

In GWO, the search behavior is performed under the guidance of the three leading wolves, wolves α, β, and δ, and the weights of the three leaders (one-third each) are the same in the updating formula. This situation does not allow prioritization of the most important, which is not consistent with the natural behavior of wolves (Sahoo and Panda, 2018; Gupta and Deep, 2020). To better reflect the leadership of wolves α, β, and δ, several improved GWOs have been proposed which modify the updating mechanism of the search wolf. The hierarchy of the pyramid (WA-GWO), the hierarchy of a weighting based on the fitness of α, β, and δ wolves (WW-GWO), and the hierarchy of the dynamic fuzzy weight (FW-GWO) were applied to modify GWO (Rodríguez et al., 2017). Some different selection schemes have been proposed to optimize GWO, including a proportion method, championship method, universal sampling method, linear rank method, and random selection method (Al-Betar et al., 2018). The exploration-enhanced GWO (EEGWO) was also proposed (Long et al., 2018), in which a new position updating equation was presented by applying a random individual.

The centralized hunting under the leadership of the three leading wolves in GWO would inevitably reduce the diversity of wolves in the search process. To reduce the possibility of falling into local optima, several improved GWOs have been obtained (Gupta and Deep, 2018, 2019c). Evolutionary population dynamics was introduced, which enhances the diversity of the search wolves, and the improved GWO is called GWO-EPD (Saremi et al., 2015). A novel clustering method was used to enhance GWO, in which binomial crossover and Lévy flight steps were taken to enhance the search capability (Tripathi et al., 2018). By combining the mutation operator and the elimination-reconstruction mechanism, the diversity of the search wolves around wolf α and in the whole feasible region is increased, and the optimized algorithm is named MR-GWO (Zhang XQ and Ming, 2017).

In addition, to enhance the performance of GWO, some researchers have attempted to create hybrid GWOs by combining other intelligent algorithms (Daniel, 2018; Gupta and Deep, 2019b). A hybrid algorithm was proposed based on GWO and the artificial bee colony (ABC) algorithm (Cong et al., 2018). A mixed algorithm was obtained by combining biogeography-based optimization and the grey wolf optimizer (Zhang XM et al., 2018). The combination of the ant lion algorithm and GWO was used to solve the complex feature selection problem in machine learning (Zawbaa et al., 2018). The sine cosine algorithm (SCA) has also been combined with GWO to maintain balance between exploration and development (Gupta et al., 2020).

Different strategies have been adopted and different variants have been obtained in the above studies to address the different shortcomings of GWO. All the above GWO variants are developed based on the standard GWO, never considering a change in the GWO algorithm flow. In the GWO flow, the search wolf cannot update its position before all search wolves complete the comparison with the leading wolves, so the current search wolf's position is not promptly updated, which is not conducive to the rapid convergence of the algorithm. Based on this, two dynamic GWO structures are proposed in this paper, and the performances of other improved GWOs based on the dynamic GWO structures are discussed.

2 Static grey wolf optimizer

In GWO, a pyramid model of four levels is proposed to simulate the hunting behavior of the grey

wolf (Fig. 1). Wolf α is the highest leader, wolf β is second only to wolf α, and wolf δ is second to wolf β. The number of the leading wolves, wolf α, β, or δ, is always set to 1. Wolf ω is the lowest-ranking wolf in GWO, often referred to as the search wolf, and the hunting behavior is done mainly by wolf ω in GWO. In the optimization application, the search for prey is the search for the optimal value. In each iteration of GWO, all the search wolves must update their positions, and the optimization information must be saved; wolf α is used to save the optimal value, wolf β the second-best value, and wolf δ the third-best value. The position of the search wolf is updated by Eqs. (1)–(7). These equations indicate the position of the search wolf, which is updated based on the guidance of the three leading wolves.

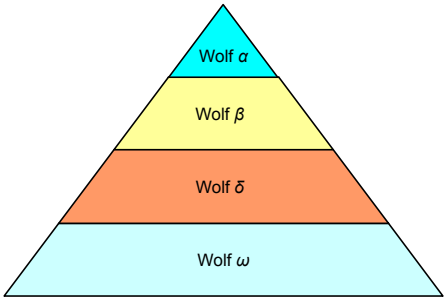

Fig. 1 The pyramid model of four levels

$$\boldsymbol{D}_\alpha = \left|\boldsymbol{C}_1 \cdot \boldsymbol{X}_\alpha - \boldsymbol{X}(t)\right|, \tag{1}$$

$$\boldsymbol{D}_\beta = \left|\boldsymbol{C}_2 \cdot \boldsymbol{X}_\beta - \boldsymbol{X}(t)\right|, \tag{2}$$

$$\boldsymbol{D}_\delta = \left|\boldsymbol{C}_3 \cdot \boldsymbol{X}_\delta - \boldsymbol{X}(t)\right|, \tag{3}$$

$$\boldsymbol{X}_1(t+1) = \boldsymbol{X}_\alpha - \boldsymbol{A}_1 \cdot \boldsymbol{D}_\alpha, \tag{4}$$

$$\boldsymbol{X}_2(t+1) = \boldsymbol{X}_\beta - \boldsymbol{A}_2 \cdot \boldsymbol{D}_\beta, \tag{5}$$

$$\boldsymbol{X}_3(t+1) = \boldsymbol{X}_\delta - \boldsymbol{A}_3 \cdot \boldsymbol{D}_\delta, \tag{6}$$

$$\boldsymbol{X}(t+1) = [\boldsymbol{X}_1(t+1) + \boldsymbol{X}_2(t+1) + \boldsymbol{X}_3(t+1)]/3. \tag{7}$$

In Eqs. (1)–(7), t denotes the current iteration, $\boldsymbol{X}(t)$ is the current position vector of the search wolf, $\boldsymbol{X}(t+1)$ is the updated position vector of the search wolf, $\boldsymbol{X}_\alpha$ is the position vector of wolf α, $\boldsymbol{X}_\beta$ is the position vector of wolf β, $\boldsymbol{X}_\delta$ is the position vector of wolf δ, and the general formulae of $\boldsymbol{A}$ and $\boldsymbol{C}$ are

$$\boldsymbol{A} = 2\boldsymbol{r}_1 \cdot \boldsymbol{a} - \boldsymbol{a}, \tag{8}$$

$$\boldsymbol{C} = 2 \cdot \boldsymbol{r}_2. \tag{9}$$

Here, the elements of $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are random numbers in [0, 1], and the value of parameter $\boldsymbol{a}$ changes from 2 to 0 as the number of iterations increases; thus, the maximum range of elements in matrix $\boldsymbol{A}$ is [−2, 2].

Fig. 2 is the GWO flow chart. In Fig. 2, the superscript "t" means that its corresponding parameters are related only to the current iteration, the superscript "t+1" means that its corresponding parameters are related to the next iteration, and the subscript "i" means the i^{th} search wolf. f_i^t is the fitness value of the i^{th} search wolf in the current iteration. f_α is the fitness value of wolf α. f_β is the fitness value of wolf β. f_δ is the fitness value of wolf δ. As can be seen from Fig. 2, except the large loop of one iteration, there are two small loops in each iteration. One loop compares the fitness value of the search wolf with f_α, f_β, and f_δ, to update wolves α, β, and δ, and the other loop is to update the position of the search wolf in turn.

In the GWO flow chart, the fitness value of the search wolf is calculated first. Then the fitness value will be compared with the values of wolves α, β, and δ to determine the better wolf α, β, or δ. After all the comparisons with the search wolves, the best positions of wolves α, β, and δ will be determined and applied in Eq. (7) to update the position of the search wolves one by one. When all search wolves complete their position updates, one iteration ends. One iteration after one iteration, the GWO algorithm does not stop until the end of iteration condition is satisfied. Finally, the optimal solution to the problem is obtained from wolf α.

In each GWO iteration, search wolves cannot update their positions until all search wolves have completed the comparisons with wolves α, β, and δ. In other words, even though one search wolf has completed comparisons with wolves α, β, and δ, its position cannot be updated and it must wait for some time. Therefore, we call the standard GWO the static GWO. The static GWO has the following characteristics: (1) Use the same wolves α, β, and δ to update the positions of all search wolves. (2) The position updating of the search wolf must wait for other search wolves to complete their comparisons with wolves α, β, and δ. (3) The positions of wolves α, β, and δ applied in Eqs. (1)–(7) to update the positions of the search wolves are the best positions in the current iteration, which is the advantage of the static GWO.

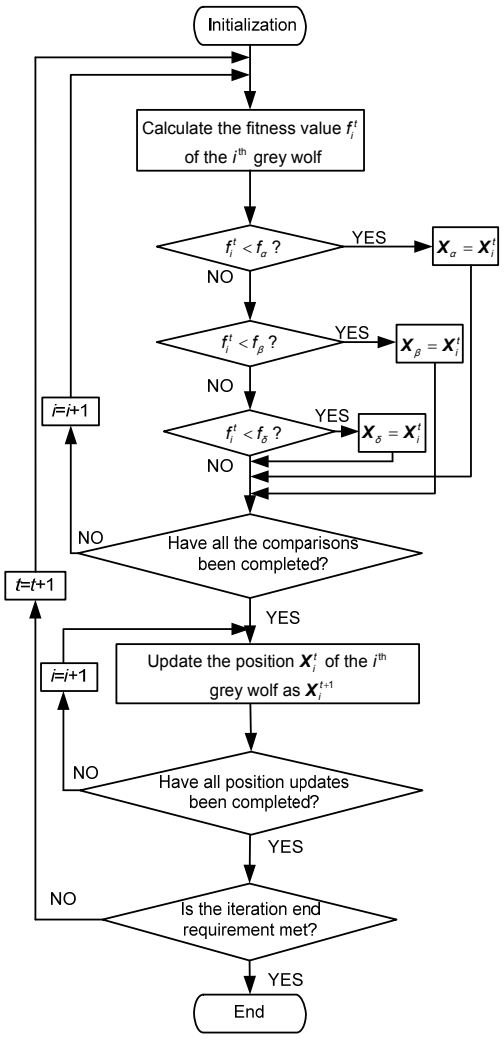

Fig. 2 Flow chart of the static grey wolf optimizer

If the whole processing of a search wolf is regarded as a calculation, the total computational amount is $N_p \times G_{max}$ in GWO, when the number of search wolves is N_p and the maximum number of iterations is G_{max}. The whole processing of a search wolf includes its fitness calculation, the comparison with wolves α, β, and δ, and its position update.

3 Dynamic grey wolf optimizer

To reduce the wait time for the position update of the search wolf, two dynamic GWOs are proposed, the first dynamic grey wolf optimizer (DGWO1) and the second dynamic grey wolf optimizer (DGWO2). The flow charts of DGWO1 and DGWO2 are shown in Figs. 3 and 4, respectively, where f_i^{t+1} is the fitness value of the i^{th} search wolf in the next iteration, and other parameters have the same meaning as in Fig. 2.

As shown in Figs. 3 and 4, if only the program language description is considered, excepting the statement "Have all the comparisons been completed?", all the other language descriptions are the same in the static GWO, DGWO1, and DGWO2. In DGWO1 and DGWO2, in addition to the external iteration loop, there is only one loop inside each iteration. The static GWO cannot update the position of each search wolf until all search wolves complete the comparisons with wolves α, β, and δ. In DGWO1 and DGWO2, things are a little different.

3.1 First dynamic GWO

In DGWO1, the current fitness value f_i^t of the i^{th} search wolf is calculated first, and then after comparing f_i^t with f_α, f_β, and f_δ, the positions and the fitness values of the three leading wolves, $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, $\boldsymbol{X}_\delta$, f_α, f_β, and f_δ, may be updated. Next, the current position of the i^{th} search wolf will be updated as $\boldsymbol{X}_i^{t+1}$ by Eqs. (1)–(7) with the $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ newly conformed. Then do the same work for the $(i+1)^{th}$ search wolf to obtain the updated position $\boldsymbol{X}_{i+1}^{t+1}$ until the position of the last search wolf is updated. Thus, one iteration ends. When all iterations are done, the DGWO1 algorithm will stop. In DGWO1, the position of the search wolf will be promptly updated with the $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ newly conformed after its current fitness value f_i^t is compared with those of f_α, f_β, and f_δ. The position updating of the search wolf does not need to wait for other fitness values to be compared with f_α, f_β, and f_δ, so the convergence of the algorithm could be accelerated to a certain extent. After the analysis of the flow chart, it is also found that the values of $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ applied to update the i^{th} search wolf in Eqs. (1)–(7) may be different from the ones applied to update the $(i+1)^{th}$ search wolf. That is, in the same iteration, for the update of different search wolves different $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ may be used.

If the whole processing of a search wolf is also regarded as a calculation, the same as the static GWO, then the total computational amount is also $N_p \times G_{max}$ in DGWO1, when the number of search wolves is N_p and the maximum number of iterations is G_{max}.

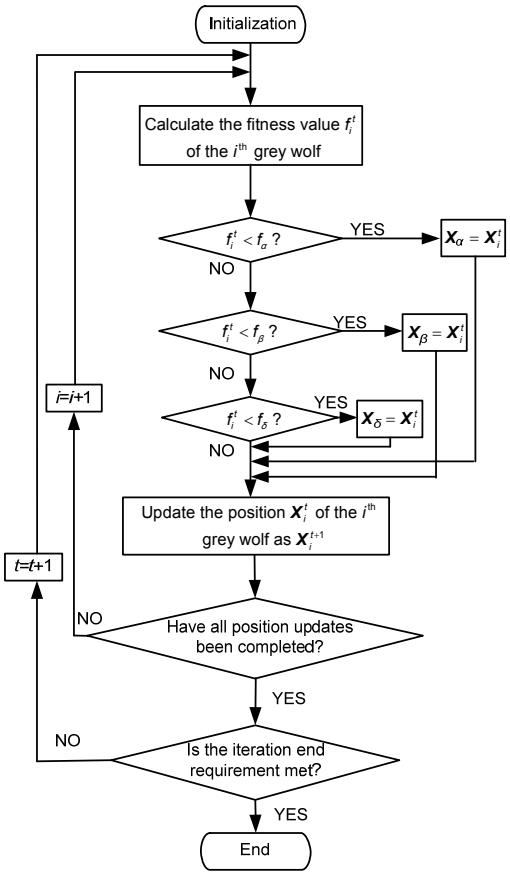

Fig. 3 Flow chart of the first dynamic grey wolf optimizer

The features of the DGWO1 are as follows:

(1) The position updating of each search wolf does not need to wait for other search wolves.

(2) The $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ may be different for each search wolf in the position updating. In addition, it should be noted that the values of $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ applied for the position update of the search wolf may not be the best ones in this iteration, but they are certainly the best values in the current search.

Just for its first feature, the DGWO1 is called the "dynamic" GWO. "Dynamic" means that there is no waiting for other search wolves to be compared with the leader wolves, and the latest $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ can be applied in time to calculate the position update of the current search wolf.

The steps of DGWO1 are as follows:

Step 1: Initialize the wolf pack and its parameters. Initialize the positions of all wolves, including wolves α, β, δ, and all ω search wolves, and give the initial fitness values of wolves α, β, and δ, respectively.

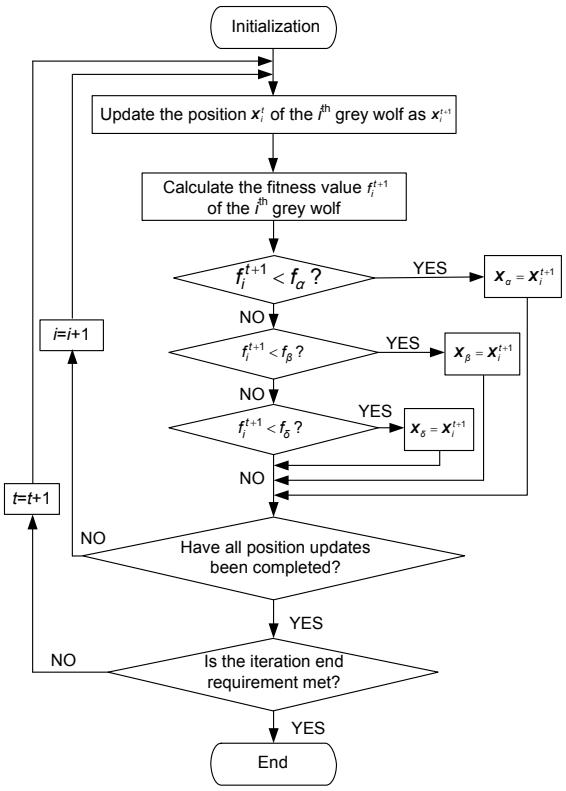

Fig. 4 Flow chart of the second dynamic grey wolf optimizer

Step 2: Calculate the fitness value f_i^t of the i^{th} search wolf, and compare it with f_α, f_β, and f_δ. If f_i^t is better, replace the position of wolf α, β, or δ with $\boldsymbol{X}_i^t$, and replace f_α, f_β, or f_δ with f_i^t.

Step 3: The search wolf encircles and hunts the prey under the leadership of wolves α, β, and δ, and it will move around wolves α, β, and δ following Eqs. (1)–(7).

Step 4: Check if the updates of all search wolves have been completed. If so, go to step 5; otherwise, return to step 2.

Step 5: If the iteration end requirement is met, the algorithm is stopped; otherwise, return to step 2.

3.2 Second dynamic GWO

Fig. 4 shows the flow chart of DGWO2. In the DGWO2, the position $\boldsymbol{X}_i^t$ of the i^{th} search wolf is first updated to $\boldsymbol{X}_i^{t+1}$ with Eqs. (1)–(7), in which $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ are obtained after comparing the $(i-1)^{th}$ search wolf with wolves α, β, and δ. Then the fitness value of the i^{th} updated search wolf is calculated, which will be compared with f_α, f_β, or f_δ respectively to obtain better

$\boldsymbol{X}_\alpha$ and f_α, $\boldsymbol{X}_\beta$ and f_β, or $\boldsymbol{X}_\delta$ and f_δ, preparing for the $(i+1)^{\text{th}}$ search wolf. Looping in the same way as above until all the search wolves finish the corresponding processing, an iteration ends. If the end requirement is satisfied, the algorithm can stop; otherwise, the next iteration will be done.

In DGWO2, there is also no need to wait for the comparison of other search wolves with wolves α, β, and δ. The "dynamic" part of DGWO2 is also reflected in the timely application of the newly obtained $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ for the position update of the search wolf. The difference between DGWO2 and DGWO1 is that the values of $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ for the position update of the i^{th} search wolf in DGWO1 are those obtained after comparing the i^{th} search wolf and wolves α, β, and δ, whereas the values of $\boldsymbol{X}_\alpha$, $\boldsymbol{X}_\beta$, and $\boldsymbol{X}_\delta$ for the position update of the i^{th} search wolf in DGWO2 are those obtained after comparing the $(i-1)^{\text{th}}$ search wolf and wolves α, β, and δ.

If the whole processing of a search wolf is also regarded as a calculation, then when the number of search wolves is N_{p} and the maximum number of iterations is $G_{\max}$, the total computational amount is also $N_{\text{p}}\times G_{\max}$ in DGWO2, the same as in DGWO1 and GWO.

4 Experimental test

4.1 Introduction to test functions

To test the performance of the proposed dynamic GWOs, the functions of CEC2014 are selected here, which can be found at http://ieee-wcci2014.org/. Table 1 gives a brief introduction of each function.

4.2 Experimental results

The experimental results of GWO, DGWO1, and DGWO2 applied to 30 standard test functions of CEC2014 are shown in Table 2. In all experiments,

Table 1 Introduction of CEC2014 test functions

Category	No.	Function description	Range	Dimension	$f_{\min}$
Unimodal functions	Fun1	Rotated high conditioned elliptic function	[−100, 100]D	30	100
	Fun2	Rotated bent cigar function	[−100, 100]D	30	200
	Fun3	Rotated discus function	[−100, 100]D	30	300
Simple multimodal functions	Fun4	Shifted and rotated Rosenbrock's function	[−100, 100]D	30	400
	Fun5	Shifted and rotated Ackley's function	[−100, 100]D	30	500
	Fun6	Shifted and rotated Weierstrass function	[−100, 100]D	30	600
	Fun7	Shifted and rotated Griewank's function	[−100, 100]D	30	700
	Fun8	Shifted Rastrigin's function	[−100, 100]D	30	800
	Fun9	Shifted and rotated Rastrigin's function	[−100, 100]D	30	900
	Fun10	Shifted Schwefel's function	[−100, 100]D	30	1000
	Fun11	Shifted and rotated Schwefel's function	[−100, 100]D	30	1100
	Fun12	Shifted and rotated Katsuura's function	[−100, 100]D	30	1200
	Fun13	Shifted and rotated happy cat function	[−100, 100]D	30	1300
	Fun14	Shifted and rotated HGBat function	[−100, 100]D	30	1400
	Fun15	Shifted and rotated expanded Griewank's plus	[−100, 100]D	30	1500
	Fun16	Shifted and rotated expanded Scaffer's F6 function	[−100, 100]D	30	1600
Hybrid functions	Fun17	Hybrid function 1 (N=3)	[−100, 100]D	30	1700
	Fun18	Hybrid function 2 (N=3)	[−100, 100]D	30	1800
	Fun19	Hybrid function 3 (N=4)	[−100, 100]D	30	1900
	Fun20	Hybrid function 4 (N=4)	[−100, 100]D	30	2000
	Fun21	Hybrid function 5 (N=5)	[−100, 100]D	30	2100
	Fun22	Hybrid function 6 (N=5)	[−100, 100]D	30	2200
Composition functions	Fun23	Composition function 1 (N=5)	[−100, 100]D	30	2300
	Fun24	Composition function 2 (N=3)	[−100, 100]D	30	2400
	Fun25	Composition function 3 (N=3)	[−100, 100]D	30	2500
	Fun26	Composition function 4 (N=5)	[−100, 100]D	30	2600
	Fun27	Composition function 5 (N=5)	[−100, 100]D	30	2700
	Fun28	Composition function 6 (N=5)	[−100, 100]D	30	2800
	Fun29	Composition function 7 (N=3)	[−100, 100]D	30	2900
	Fun30	Composition function 8 (N=3)	[−100, 100]D	30	3000

Table 2 The results of GWO, DGWO1, and DGWO2

No.	Algorithm	E_{ave}	E_{min}	E_{max}	STD	T_{ave} (s)
	GWO	7.7348E+07	1.7351E+07	2.2245E+08	5.4443E+07	0.620 000 00
Fun1	DGWO1	6.9518E+07	1.0246E+07	1.6806E+08	4.3795E+07	0.634 266 67
	DGWO2	9.7561E+07	3.0570E+07	3.6159E+08	6.4678E+07	0.708 833 33
	GWO	1.7374E+09	3.2964E+08	7.1383E+09	1.5693E+09	0.576 000 00
Fun2	DGWO1	9.3054E+08	8.2950E+07	3.0748E+09	7.1292E+08	0.608 600 00
	DGWO2	1.5497E+09	1.3502E+08	4.5133E+09	1.1434E+09	0.647 700 00
	GWO	4.4154E+04	2.1225E+04	7.2421E+04	1.2688E+04	0.580 500 00
Fun3	DGWO1	4.2716E+04	1.1950E+04	6.7400E+04	1.1629E+04	0.600 133 33
	DGWO2	4.2887E+04	1.9757E+04	5.6139E+04	9.3738E+03	0.636 700 00
	GWO	272.776 001	111.389 438	991.139 518	161.183 792	0.606 066 67
Fun4	DGWO1	249.367 799	138.746 704	389.764 796	63.725 696 6	0.598 933 33
	DGWO2	269.071 525	135.750 025	468.485 451	77.771 601	0.653 233 33
	GWO	21.048 390 7	20.954 732 4	21.119 694 2	0.046 445 16	0.598 133 33
Fun5	DGWO1	21.073 439 0	20.923 770 9	21.168 280 9	0.059 163 18	0.616 200 00
	DGWO2	21.052 303 0	20.930 537 3	21.150 743 5	0.049 953 19	0.671 100 00
	GWO	16.138 581 7	7.719 749 90	22.520 341 4	2.720 183 57	2.186 666 67
Fun6	DGWO1	13.345 383 1	7.793 407 07	18.014 303 4	2.856 747 81	2.193 333 33
	DGWO2	16.046 175 9	10.906 464 3	22.160 307 9	3.139 867 85	2.371 666 67
	GWO	18.677 730 1	2.290 059 49	50.470 529 5	12.598 917 0	0.608 166 67
Fun7	DGWO1	12.119 935 7	4.077 443 55	46.909 334 9	9.887 208 00	0.624 766 67
	DGWO2	16.724 520 7	3.442 175 71	42.524 885 9	11.370 005 4	0.706 066 67
	GWO	84.128 305 8	51.135 448 8	119.946 435	18.667 089 5	0.568 033 33
Fun8	DGWO1	83.216 554 1	45.237 634 3	183.901 163	29.200 958 1	0.586 800 00
	DGWO2	96.119 267 6	46.227 783 2	157.915 940	25.803 891 5	0.678 100 00
	GWO	101.913 738	53.687 965 8	174.292 255	28.048 629 9	0.591 533 33
Fun9	DGWO1	107.711 217	66.619 405 0	247.011 399	45.636 299 6	0.610 900 00
	DGWO2	104.673 026	76.026 141 4	155.802 704	20.196 713 9	0.670 200 00
	GWO	2882.950 67	1876.128 08	5506.288 99	742.515 951	0.641 900 00
Fun10	DGWO1	2295.958 83	1188.735 45	3673.702 85	608.441 075	0.665 166 67
	DGWO2	2786.682 12	1750.041 73	4119.279 84	627.442 287	0.768 700 00
	GWO	3663.280 72	2142.506 64	7202.166 16	1288.925 83	0.662 566 67
Fun11	DGWO1	3563.578 79	1985.361 06	7227.054 04	1103.213 96	0.685 666 67
	DGWO2	3337.252 81	2283.152 95	5862.003 51	687.189 136	0.821 766 67
	GWO	2.985 698 07	0.328 392 73	3.822 384 31	0.768 025 50	0.860 966 67
Fun12	DGWO1	2.412 137 13	0.175 209 87	3.783 569 46	1.195 294 87	0.896 033 33
	DGWO2	2.803 669 40	0.261 362 94	3.990 728 62	0.988 940 55	1.053 166 67
	GWO	0.536 126 93	0.345 703 55	1.501 063 15	0.198 913 68	0.574 966 67
Fun13	DGWO1	0.494 892 44	0.332 262 18	0.894 971 50	0.111 282 74	0.596 166 67
	DGWO2	0.524 429 99	0.306 659 44	0.806 768 69	0.123 578 13	0.704 866 67
	GWO	4.809 309 63	0.204 567 04	24.518 411 7	6.343 336 27	0.576 400 00
Fun14	DGWO1	1.056 274 75	0.227 968 90	10.467 584 7	1.863 233 08	0.601 733 33
	DGWO2	2.506 442 19	0.226 079 42	12.361 006 2	3.795 511 21	0.700 600 00
	GWO	315.635 614	6.875 791 12	4741.164 01	991.317 380	0.594 800 00
Fun15	DGWO1	48.509 007	15.957 130 1	349.030 677	75.821 151 9	0.615 533 33
	DGWO2	191.716 911	11.706 372 4	1988.790 37	480.383 755	0.730 166 67
	GWO	12.158 315	11.088 895 5	13.068 278 0	0.610 878 21	0.600 533 33
Fun16	DGWO1	11.736 275 3	9.903 901 14	12.840 498 9	0.781 347 39	0.622 533 33
	DGWO2	11.936 781 4	10.277 638 8	12.809 454 2	0.598 443 99	0.747 300 00
	GWO	2.8962E+06	2.7577E+05	1.4475E+07	3.0274E+06	0.629 633 33
Fun17	DGWO1	2.2905E+06	1.8796E+05	7.6651E+06	1.9662E+06	0.648 833 33
	DGWO2	2.7561E+06	1.6183E+05	1.1978E+07	3.0029E+06	0.773 766 67

To be continued

Table 2

No.	Algorithm	E_{ave}	E_{min}	E_{max}	STD	T_{ave} (s)
	GWO	1.3223E+07	8.5780E+03	7.5837E+07	1.9894E+07	0.593 100 00
Fun18	DGWO1	8.4606E+06	1.0638E+04	6.2757E+07	1.8511E+07	0.618 433 33
	DGWO2	1.3929E+07	9.4177E+02	1.0145E+08	2.6621E+07	0.743 300 00
	GWO	50.6284625	12.306 446 8	92.794 2614	26.651 3327	0.917 233 33
Fun19	DGWO1	32.8742287	12.792 560 1	76.307 1264	20.998 6305	0.937 533 33
	DGWO2	45.0406580	16.218 304 2	88.189 0767	26.065 4346	1.122 966 67
	GWO	2.9684E+04	1.4239E+04	7.3322E+04	1.4796E+04	0.597 400 00
Fun20	DGWO1	3.2474E+04	3.3878E+03	8.8334E+04	2.0072E+04	0.615 100 00
	DGWO2	3.4576E+04	8.8262E+03	9.5146E+04	2.2123E+04	0.741 200 00
	GWO	9.2300E+05	2.8342E+04	1.0654E+07	1.8986E+06	0.620 233 33
Fun21	DGWO1	8.2583E+05	7.6834E+04	1.0495E+07	1.8703E+06	0.640 833 33
	DGWO2	1.5589E+06	1.1307E+05	1.0072E+07	2.2829E+06	0.752 633 33
	GWO	427.903 188	84.751 731	1144.203 63	215.941 522	0.652 233 33
Fun22	DGWO1	348.052 763	58.330 618	604.669 184	153.379 408	0.670 466 67
	DGWO2	457.596 281	44.498 0107	1018.284 01	246.024 009	0.755 133 33
	GWO	338.382 543	325.772 425	360.410 310	9.336 105 09	0.934 933 33
Fun23	DGWO1	337.020 531	325.456 587	365.686 879	11.214 597 4	0.951 666 67
	DGWO2	339.492 910	326.283 158	371.444 432	12.179 159 5	1.128 633 33
	GWO	200.056 117	200.033 488	200.109 925	0.019 251 47	0.821 700 00
Fun24	DGWO1	200.085 139	200.050 013	200.141 596	0.024 645 05	0.848 333 33
	DGWO2	200.034 313	200.021 036	200.064 168	0.011 843 02	0.971 133 33
	GWO	212.157 269	200	220.079 676	5.512 285 26	0.890 933 33
Fun25	DGWO1	212.741 401	200	220.619 928	5.162 735 19	0.911 066 67
	DGWO2	214.008 045	200	222.217 401	4.026 295 09	0.996 566 67
	GWO	127.378 859	100.325 550	200.281 093	44.634 295 8	2.609 166 67
Fun26	DGWO1	140.398 082	100.348 975	200.382 863	49.631 866 5	2.629 900 00
	DGWO2	160.293 829	100.291 873	200.558 498	49.610 911 6	2.952 700 00
	GWO	645.187 725	428.495 181	871.723 890	137.250 692	2.574 866 67
Fun27	DGWO1	617.996 603	420.905 577	812.234 452	126.461 387	2.592 800 00
	DGWO2	727.147 534	409.297 087	926.558 827	115.610 516	2.866 166 67
	GWO	1168.927 28	887.657 395	2205.104 65	299.722 556	1.058 233 33
Fun28	DGWO1	1083.662 76	856.961 974	1535.041 87	169.773 424	1.079 166 67
	DGWO2	1127.608 03	880.408 681	1800.382 09	251.710 203	1.172 133 33
	GWO	1.4705E+06	6.1953E+03	1.8575E+07	3.9090E+06	1.183 700 00
Fun29	DGWO1	5.6260E+04	6.1817E+03	4.0440E+05	9.2709E+04	1.199 200 00
	DGWO2	2.1456E+06	5.6005E+03	1.3420E+07	3.8727E+06	1.252 666 67
	GWO	7.0658E+04	9.2651E+03	2.9369E+05	5.5296E+04	0.884 600 00
Fun30	DGWO1	6.2713E+04	1.7721E+04	1.4785E+05	3.5613E+04	0.907 200 00
	DGWO2	6.7740E+04	1.3471E+04	1.5801E+05	3.4178E+04	0.960 800 00

the number of search wolves is set as 50, and the maximum number of iterations is set as 500. For each function, the optimization experiment based on each algorithm is repeated 30 times. E_{ave} in Table 2 is the average value of 30 experimental errors, E_{min} is the minimum error, E_{max} is the maximum error, STD is the average standard deviation, and T_{ave} is the mean runtime of each experiment.

4.3 Analysis and discussion

From the experimental results in Table 2, comparing DGWO1 with GWO, there are 24 functions whose average DGWO1 errors E_{ave} are superior to the GWO errors, 17 functions whose minimum DGWO1 errors are better than the GWO errors, 21 functions whose maximum DGWO1 errors are lower than the GWO errors, and 20 functions whose DGWO1 average standard deviations are less than the GWO average standard deviations. All of these numbers above are more than half of 30, which can show that the overall optimization capability of DGWO1 is better than that of GWO. Comparing DGWO2 with GWO, there are 17 average errors, 17 minimum errors, 20 maximum errors, and 19 standard deviations of

DGWO2, which are superior to the ones of GWO, respectively. These numbers are also more than half of 30, which also shows that the overall optimization ability of DGWO2 is better than that of GWO. The logarithms of the average errors of GWO, DGWO1, and DGWO2 are shown in Fig. 5. From the average errors, the improved effect of DGWO1 is more noticeable than that of DGWO2.

According to the T_{ave} in Table 2, the mean runtimes of DGWO1 and DGWO2 are almost longer than that of GWO. However, as explained earlier, if the static GWO, DGWO1, and DGWO2 have the same total computational amount, why are there differences in the runtimes in the course of the experiments? Note that the total computational amount described above can only roughly reflect the number of cycles that the algorithm should perform, but there are many selected statements in the cycle structure. The selected statements about the fitness comparisons between each search wolf and wolves α, β, and δ are the most likely to be executed repeatedly. The characteristic of the selected statement is that if its condition is satisfied, get inside the selected statement or skip it. Thus, the whole runtime of the algorithm will be affected by whether the condition of the selected statement is satisfied. The more chance there is to get inside the selected statement, the larger the runtime that will be needed.

From the analysis of the dynamic GWOs, it is known that the newly updated positions of wolves α, β, and δ will be applied promptly to calculate the updated position of the search wolf to make the updated position better. Therefore, the positions of wolves α, β, and δ are updated more frequently. The increase of the number of selected statements in the optimization means that the possibility of getting inside the process and the total runtimes of the dynamic GWOs are increased.

In DGWO2, the position update of the i^{th} search wolf can be impacted to some extent by the position of the $(i-1)^{th}$ search wolf. This means that in an iteration the position of the first search wolf can affect the position of the second search wolf, the position of the second search wolf can affect the position of the third wolf, the position of the third wolf can affect the position of the fourth wolf, etc. Thus, the former and the latter search wolves influence each other, strengthening the communication between the wolves to be helpful in finding the optimal value of the problem, which is different from the static GWO. In DGWO2, the topology of the former and the latter influencing each other is good for determining the best wolf, and it may lead to the position of wolf α, β, or δ being updated in every iteration; that is, there would be many chances to get inside the selected statement in the process of comparing the fitness of the search wolf and the leader wolves, which would increase the runtime of the experiment. Maybe that is why the runtime of the DGWO2 is longer than the others in Table 2.

To be clear, the high update probability of the position of wolf α, β, or δ in each iteration does not mean that the final result must be the best one. In these experiments, the results of DGWO2 are slightly worse than the ones of DGWO1, probably because of the way the former and the latter influence each other in

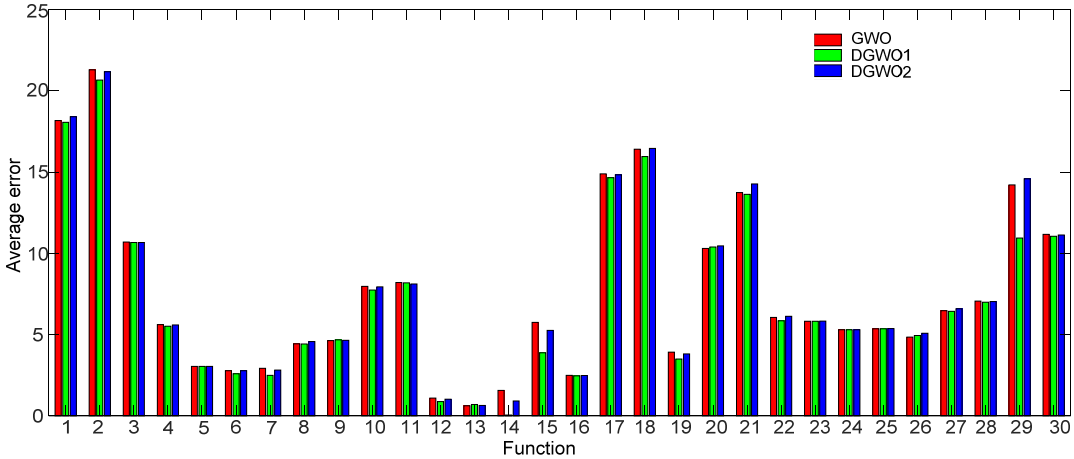

Fig. 5 The average errors (in log) of GWO, DGWO1, and DGWO2

making the step-by-step convergence slow down the optimization speed of DGWO2.

The convergence graphs of some functions are shown in Fig. 6. According to the convergence graphs, the convergence speeds of DGWO1 and DGWO2 are better than those of GWO under the same number of iterations.

5 Improved strategies with dynamic GWOs

Some improved GWO algorithms have been mentioned in the introduction, all of which are obtained based on the static GWO. Here, the dynamic GWOs will be applied to these improved GWO algorithms, using MR-GWO, β-GWO, and GWO-EPD as examples. For a systematic study of the novel improved dynamic GWOs, the following rules are created.

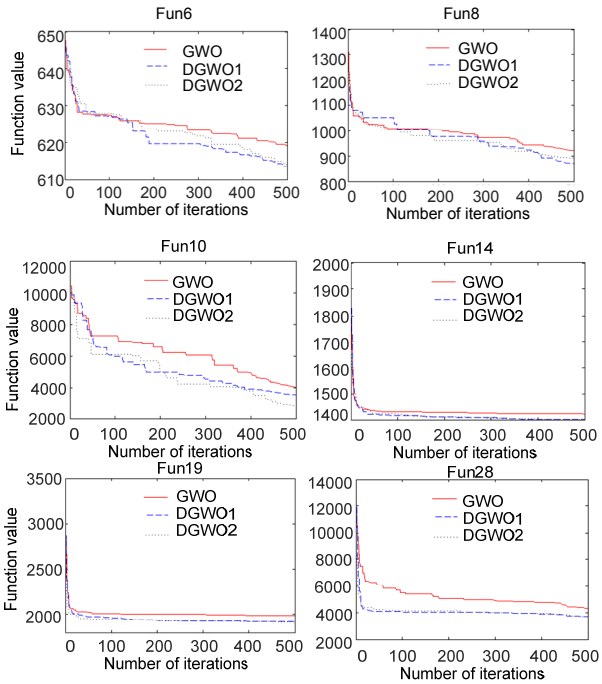

Fig. 6 The convergence graphs of some functions

If the modified algorithm is based on the dynamic GWO structure, a letter "D" is added before the original abbreviated name of the improved GWO based on the static GWO. For example, if the improvement in MR-GWO applies to the dynamic GWO, the novel improved algorithm can be noted as "DMR-GWO." If the modification is based on the first dynamic GWO structure, add the number "1" after the original abbreviated name; if the modification is based on the second dynamic GWO structure, add the number "2" after the original abbreviated name of the algorithm.

According to this convention, in MR-GWO, if the improvements over GWO are shifted to the first dynamic GWO, the novel improved algorithm would be called DMR-GWO1, and if the improvements are shifted to the second dynamic GWO, the novel improved algorithm will be called DMR-GWO2. Similarly, there are Dβ-GWO1, Dβ-GWO2, DGWO-EPD1, and DGWO-EPD2.

The functions of CEC2014 are still selected as the optimized objects. Here, because there are many algorithms studied, to reduce the length of the table header, the further shorthand notations of each algorithm are listed in Table 3. The experimental results are shown in Table 4, and the data in Table 4 are the average errors of 30 repeated experiments of each function based on each algorithm. Differential evolution (DE) is also introduced to compare with others.

The following is a detailed analysis and comparison of the experimental results:

For DMR-GWO1, there are 21 functions whose average errors are lower than or equal to that of MR-GWO. For DMR-GWO2, there are 16 functions whose average errors are less than that of MR-GWO. Both the numbers are greater than half of 30, which shows that it is more effective to introduce the differential mutation operator and the elimination-recombination mechanism to the dynamic GWOs than to the static GWO.

Table 3 Further shorthand notations for each algorithm

Algorithm	Shorthand notation	Algorithm	Shorthand notation	Algorithm	Shorthand notation
GWO	A1	DMR-GWO1	DA21	Dβ-GWO2	DA32
DGWO1	DA11	DMR-GWO2	DA22	GWO-EPD	A4
DGWO2	DA12	β-GWO	A3	DGWO-EPD1	DA41
MR-GWO	A2	Dβ-GWO1	DA31	DGWO-EPD2	DA42

For Dβ-GWO1, there are 17 functions whose average errors are smaller than that of β-GWO. For Dβ-GWO2, there are 14 functions whose average errors are smaller than that of β-GWO. This shows that it works better to introduce β-chaotic sequences into parameter $\boldsymbol{a}$ for the first dynamic GWO than for the static GWO.

For DGWO-EPD1, there are 19 functions whose average errors are lower than or equal to that of GWO-EPD. For DGWO-EPD2, there are 16 functions whose average errors are smaller than that of GWO-EPD. Same as above, this also shows that it is better to introduce the evolutionary population dynamics to the dynamic GWOs than to the static GWO. The novel improved algorithms based on the dynamic GWOs are more competitive in the optimization.

The number of the best results obtained by each algorithm is counted as follows: DGWO2 has one best result, MR-GWO has 10 best results (including seven results equaling the ones of DMR-GWO1), DMR-GWO1 has 10 best results, DMR-GWO2 has 10 best results, Dβ-GWO1 has two best result, GWO-EPD has five best results (including four results equaling the ones of DMR-GWO1), DGWO-EPD1 has four best results (all the results equaling the ones of DMR-GWO1), and DGWO-EPD2 has three best results. Fig. 7 shows the distribution of the best results in each algorithm. On one hand, the best results are

Table 4 Results of the improved dynamic GWO algorithms

No.	A1	DA11	DA12	A2	DA21	DA22	A3	DA31	DA32	A4	DA41	DA42	DE
1	7.73E7	6.95E7	9.76E7	3.06E7	3.21E7	**2.57E7**	7.58E7	6.69E7	6.94E7	3.84E7	3.32E7	2.88E7	2.25E9
2	1.74E9	9.31E8	1.55E9	4.27E7	4.55E7	**8.65E6**	1.98E9	1.37E9	1.89E9	3.52E7	3.43E7	1.24E7	9.8E10
3	4.42E4	4.27E4	4.29E4	2.33E4	**2.29E4**	2.72E4	4.08E4	4.53E4	4.49E4	2.89E4	2.97E4	2.94E4	1.31E7
4	272.776	249.368	269.072	180.607	177.784	**143.461**	271.894	247.689	246.676	179.890	191.136	158.569	2.34E4
5	21.0484	21.0734	21.0523	**20.8545**	20.8944	21.0569	21.0617	21.0696	21.0536	21.0072	20.9860	21.0342	21.02
6	16.1386	13.3454	16.0462	20.2130	20.3455	**12.4589**	15.5060	14.4338	16.9081	17.0944	17.0788	14.9227	47.38
7	18.6777	12.1199	16.7245	1.430 85	1.406 07	1.15631	14.0324	11.1091	17.1134	1.281 39	1.290 83	**1.135 52**	972.63
8	84.1283	83.2166	96.1193	127.920	117.273	**76.6245**	82.9077	76.9432	87.9166	99.1351	99.2870	82.7748	404.46
9	101.914	107.711	104.673	161.362	137.883	89.3886	107.246	**87.9287**	109.793	114.559	102.408	99.4011	379.73
10	2882.95	2295.96	2786.68	3013.81	3015.27	**1996.75**	2705.92	2595.62	2877.46	2445.56	2586.64	3082.61	8.80E3
11	3663.28	3563.58	**3337.25**	3940.57	3875.21	4198.12	3775.18	3864.90	3509.92	3725.07	3498.25	3972.42	9.23E3
12	2.985 70	2.412 14	2.803 67	0.967 26	**0.865 71**	3.099 84	2.703 58	2.920 33	2.900 51	1.497 86	1.619 34	3.162 16	1.77
13	0.536 13	0.494 89	0.524 43	0.603 78	0.591 79	0.487 58	0.523 87	**0.452 91**	0.506 52	0.524 00	0.529 57	0.498 75	10.41
14	4.809 31	1.056 27	2.506 44	**0.297 54**	0.307 99	0.528 20	3.016 27	1.472 55	1.939 91	0.463 28	0.448 06	0.569 11	370.28
15	315.636	48.5090	191.717	20.3668	18.8064	**13.7993**	43.5207	98.1012	49.1236	18.4611	18.7740	16.8452	6.34E5
16	12.1583	11.7363	11.9368	**11.6643**	11.7568	12.0493	11.9108	12.0953	11.9483	12.0532	11.8940	12.2230	13.98
17	2.90E6	2.29E6	2.76E6	2.37E6	2.35E6	1.64E6	3.29E6	1.74E6	2.24E6	2.22E6	1.79E6	**1.38E6**	9.48E8
18	1.32E7	8.46E6	1.39E7	4.15E4	5.92E4	**2.52E4**	8.91E6	7.77E6	1.40E7	6.34E4	5.47E4	4.83E4	1.5E10
19	50.6285	32.8742	45.0407	22.1325	17.9416	**16.3372**	42.9697	50.7686	40.5305	19.8551	18.3296	17.1595	824.69
20	2.97E4	3.25E4	3.46E4	1.39E4	1.99E4	1.31E4	2.52E4	2.69E4	2.49E4	9.50E3	1.40E4	**1.25E4**	2.75E9
21	9.23E5	8.26E5	1.56E6	5.41E5	**4.78E5**	4.80E5	1.39E6	1.76E6	1.82E6	5.26E5	5.21E5	5.74E5	1.97E9
22	427.903	348.053	457.596	524.016	486.510	**347.260**	408.625	440.924	391.068	451.570	405.747	391.186	4.40E6
23	338.383	337.021	339.493	**200**	**200**	325.372	335.603	333.485	336.847	**200**	**200**	327.142	215.43
24	200.056	200.085	200.034	**200**	**200**	200.014	200.059	200.081	200.036	200.085	200.083	200.078	202.78
25	212.157	212.741	214.008	**200**	**200**	208.481	212.604	212.012	213.120	**200**	**200**	211.450	200.23
26	127.379	140.398	160.294	130.466	123.823	147.046	120.525	137.029	161.487	**120.468**	120.512	127.169	200.00
27	645.188	617.997	727.148	**200**	**200**	409.710	669.900	646.451	697.768	**200**	**200**	617.280	216.28
28	1168.93	1083.66	1127.61	**200**	**200**	1059.13	1181.14	1012.48	1176.88	**200**	**200**	978.382	219.08
29	1.47E6	5.63E4	2.15E6	**200**	**200**	1.57E4	6.50E5	7.84E5	9.63E5	1.65E5	2.25E5	1.08E6	1.79E7
30	7.07E4	6.27E4	6.77E4	**200**	**200**	3.10E4	6.71E4	4.88E4	5.32E4	4.57E4	3.77E4	2.70E4	1.20E6

The bold data are the best results of all algorithms

relatively divergent in each algorithm, which fully reflects the "no free lunch" theory. On the other hand, most of the best results can be obtained under the algorithms based on the dynamic GWOs, in which 16 best results can be obtained based on DGWO1 and 14 best results can be obtained based on DGWO2. Of course, there are four best results with repeated statistics. The more "best" results there are, the better the algorithm is. Thus, DMR-GWO1 and DMR-GWO2 are the best algorithms, and MR-GWO is the second-best algorithm.

Note that among all the best results, there is little difference between the numbers of MR-GWO, DMR-GWO1, and DMR-GWO2, while the superiority of DMR-GWO1 is obvious when MR-GWO is compared with DMR-GWO1 alone; this seems to be a contradiction between the two conclusions. Try to explore the contradiction; the results of the comparison are shown in Table 5. From Table 4 above, we know that there are only three results in functions 5, 14, and 16 of MR-GWO that are the best ones. There are seven MR-GWO results that equal the results of DMR-GWO1, all of which are the best results in all algorithms. Thus, for MR-GWO, the number of best results is 10, the same as that for DMR-GWO1 and DMR-GWO2. If the maximum number of the best results reflects that the algorithm is the best, DMR-GWO1 and DMR-GWO2 would be the best algorithms, as elaborated above, and MR-GWO would be the second-best algorithm. Table 5 shows that there are 21 functions (more than half of 30) with fewer DMR-GWO1 optimization errors than MR-GWO, which means that DMR-GWO1 is more competitive than MR-GWO in overall optimization performance. From the above analysis, both DMR-GWO1 and DMR-GWO2 have some advantages over MR-GWO. In fact, the competitiveness of the dynamic MR-GWOs lies more in the stronger stability of their optimization processes.

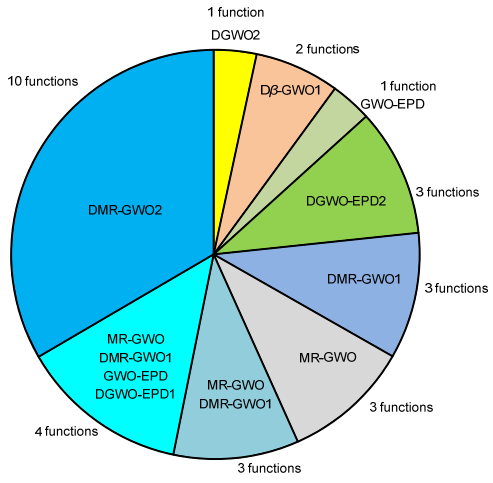

Fig. 7 Distribution of the optimal results

To sort the algorithms, the concept of rank value is given (Wu et al., 2016); this is an evaluation standard officially defined for CEC test functions.

Definition 1 For a problem, the algorithm rank is determined by the average value and the median value obtained after the maximum iteration. When there are many problems, the rank value of each algorithm is calculated by

$$\text{Rank value} = \sum_{i=1}^{n} \text{rank1}_i + \sum_{i=1}^{n} \text{rank2}_i. \qquad (10)$$

In Eq. (10), rank1_i is the average optimal value of the algorithm after multiple experiments for the i^{th} problem, and rank2_i is the median optimal value of the algorithm after multiple experiments for the i^{th} problem.

Table 6 shows the rank value results of all algorithms. The lower the rank value, the better the algorithm performance.

As can be seen from Table 6, all the improved algorithms based on DGWO1 are superior to the ones based on GWO; that is, DGWO1 is superior to GWO, DMR-GWO1 is superior to MR-GWO, Dβ-GWO1 is superior to β-GWO, and DGWO-EPD1 is superior to GWO-EPD, indicating that the improvement effect

Table 5 The detailed comparison results of MR-GWO, DMR-GWO1, and DMR-GWO2

Item	Ordinal of the function
The function corresponding to the best result in all algorithms obtained with MR-GWO	5, 14, 16, 23, 24, 25, 27, 28, 29, 30
The function whose error obtained with DMR-GWO1 is less than the one obtained with MR-GWO (Table 4)	3, 4, 7, 8, 9, 11, 12, 13, 15, 17, 19, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30
The function whose error obtained with DMR-GWO2 is less than the one obtained with MR-GWO (Table 4)	1, 2, 4, 6, 7, 8, 9, 10, 13, 15, 17, 18, 19, 20, 21, 22

based on DGWO1 is obvious. Although the rank value of DGWO2 is greater than that of GWO, and the rank value of Dβ-GWO2 is greater than that of β-GWO, the rank value of DMR-GWO2 is less than that of MR-GWO, and the rank value of DGWO-EPD2 is also less than that of GWO-EPD. Moreover, DMR-GWO2 and DGWO-EPD2 are the optimal and suboptimal algorithms among all comparison algorithms, indicating that the improvement based on DGWO2 has a certain effect.

Table 6 Rank value results of the algorithms

Algorithm	Rank value	Rank
GWO	3.82E+09	10
DGWO1	2.47E+09	8
DGWO2	4.09E+09	12
MR-GWO	1.83E+08	6
DMR-GWO1	1.48E+08	4
DMR-GWO2	7.21E+07	1
β-GWO	3.43E+09	9
Dβ-GWO1	2.21E+09	7
Dβ-GWO2	4.08E+09	11
GWO-EPD	1.55E+08	5
DGWO-EPD1	1.42E+08	3
DGWO-EPD2	1.02E+08	2
DE	2.41E+11	13

Otherwise, according to the ranking results of the rank value in Table 6, the best rank value belongs to DMR-GWO2, the second best is DGWO-EPD2, and the third and fourth best are DGWO-EPD1 and DMR-GWO1, respectively. The first four algorithms are based on the dynamic GWOs, which to some extent proves that the improvement based on the dynamic GWO algorithms is effective.

6 Conclusions

Based on the structure analysis of the standard GWO, two dynamic GWOs, DGWO1 and DGWO2, are proposed by changing the structure of the algorithm. The common point of DGWO1 and DGWO2 is that the position of the search wolf is updated in a timely manner, and it is not necessary to wait for the comparisons of all other search wolves with the three leading wolves. To reflect the timely updating of the position in DGWO1 and DGWO2, they are called the "dynamic" algorithms. The advantage of DGWO1 and DGWO2 is that the new positions of the three leading wolves can be used in time for the position update of the search wolf. Because GWO is an algorithm under the guidance of the three leading wolves, the convergence of the dynamic algorithms is faster. The difference between DGWO1 and DGWO2 is as follows: in DGWO1, the position update of the search wolf is done after the comparison between the current search wolf itself and the three leading wolves, but in DGWO2, the position update of the search wolf is done after the comparison between the previous search wolf and the three leading wolves.

From the 30 standard test function experiments of CEC2014, DGWO1 and DGWO2 are verified as superior to the static GWO. Finally, the dynamic structure is also applied to other improved GWO algorithms to obtain the corresponding improved dynamic GWO algorithms. It is also proved by these experiments that most of the improved algorithms based on the dynamic GWO structures are more effective than those based on the static GWO structure. It is also shown that DMR-GWO2 has the best performance in the 30 standard test experiments based on the rank value.

Finally, we must mention that although the advantages of the dynamic GWOs have been found from different perspectives, compared with GWO, the runtimes of both dynamic GWOs are longer under the same number of iterations for the same number of individuals. The optimization result and the runtime are often two contradictory indicators to measure an optimization algorithm. It is this contradiction that drives us to constantly strive to find better algorithms, thus promoting the continuous development of computational intelligence.

Contributors

Xiaoqing ZHANG designed the research and processed the data. Yuye ZHANG drafted the manuscript. Zhengfeng MING helped organize the manuscript. Xiaoqing ZHANG, Yuye ZHANG, and Zhengfeng MING revised and finalized the paper.

Compliance with ethics guidelines

Xiaoqing ZHANG, Yuye ZHANG, and Zhengfeng MING declare that they have no conflict of interest.

References

Al-Betar MA, Awadallah MA, Faris H, et al., 2018. Natural selection methods for grey wolf optimizer. *Expert Syst Appl*, 113:481-499. https://doi.org/10.1016/j.eswa.2018.07.022

Cong SL, Sun J, Mao HP, et al., 2018. Non-destructive detection for mold colonies in rice based on hyperspectra and GWO-SVR. *J Sci Food Agric*, 98(4):1453-1459. https://doi.org/10.1002/jsfa.8613

Daniel E, 2018. Optimum wavelet based homomorphic medical image fusion using hybrid genetic-grey wolf optimization algorithm. *IEEE Sens J*, 18(6):6804-6811. https://doi.org/10.1109/JSEN.2018.2822712

Emary E, Zawbaa HM, Hassanien AE, 2016. Binary grey wolf optimization approaches for feature selection. *Neurocomputing*, 172:371-381. https://doi.org/10.1016/j.neucom.2015.06.083

Gupta S, Deep K, 2018. Cauchy grey wolf optimiser for continuous optimisation problems. *J Exp Theor Artif Intell*, 30(6):1051-1075. https://doi.org/10.1080/0952813X.2018.1513080

Gupta S, Deep K, 2019a. Hybrid grey wolf optimizer with mutation operator. In: Bansal JC, Das KN, Nagar A, et al. (Eds.), Soft Computing for Problem Solving. Springer, Singapore, p.961-968. https://doi.org/10.1007/978-981-13-1595-4_75

Gupta S, Deep K, 2019b. A novel random walk grey wolf optimizer. *Swarm Evol Comput*, 44:101-112. https://doi.org/10.1016/j.swevo.2018.01.001

Gupta S, Deep K, 2019c. An opposition-based chaotic grey wolf optimizer for global optimisation tasks. *J Exp Theor Artif Intelll*, 31(5):751-779. https://doi.org/10.1080/0952813X. 2018.1554712

Gupta S, Deep K, 2020. A memory-based grey wolf optimizer for global optimization tasks. *Appl Soft Comput*, 93: 106367. https://doi.org/10.1016/j.asoc.2020.106367

Gupta S, Deep K, Moayedi H, et al., 2020. Sine cosine grey wolf optimizer to solve engineering design problems. *Eng Comput*, online. https://doi.org/10.1007/s00366-020-00996-y

Liu XL, Tian Y, Lei XH, et al., 2019. An improved self-adaptive grey wolf optimizer for the daily optimal operation of cascade pumping stations. *Appl Soft Comput*, 75:473-493. https://doi.org/10.1016/j.asoc.2018.11.039

Long W, Jiao JJ, Liang XM, et al., 2018. An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization. *Eng Appl Artif Intell*, 68:63-80. https://doi.org/10.1016/j.engappai.2017.10.024

Lu C, Xiao SQ, Li XY, et al., 2016. An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. *Adv Eng Softw*, 99:161-176. https://doi.org/10.1016/j.advengsoft.2016.06.004

Mirjalili S, Mirjalili SM, Lewis A, 2014. Grey wolf optimizer. *Adv Eng Softw*, 69:46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mirjalili S, Saremi S, Mirjalili SM, et al., 2016. Multi-objective grey wolf optimizer: a novel algorithm for multi-criterion optimization. *Expert Syst Appl*, 47:106-119. https://doi.org/10.1016/j.eswa.2015.10.039

Qais MH, Hasanien HM, Alghuwainem S, 2018. Augmented grey wolf optimizer for grid-connected PMSG-based wind energy conversion systems. *Appl Soft Comput*, 69: 504-515. https://doi.org/10.1016/j.asoc.2018.05.006

Rodríguez L, Castillo O, Soria J, et al., 2017. A fuzzy hierarchical operator in the grey wolf optimizer algorithm. *Appl Soft Comput*, 57:315-328. https://doi.org/10.1016/j.asoc.2017.03.048

Sahoo BP, Panda S, 2018. Improved grey wolf optimization technique for fuzzy aided PID controller design for power system frequency control. *Sustain Energy Grids Netw*, 16:278-299. https://doi.org/10.1016/j.segan.2018.09.006

Saremi S, Mirjalili SZ, Mirjalili SM, 2015. Evolutionary population dynamics and grey wolf optimizer. *Neur Comput Appl*, 26(5):1257-1263. https://doi.org/10.1007/s00521-014-1806-7

Saxena A, Kumar R, Das S, 2019. β-Chaotic map enabled grey wolf optimizer. *Appl Soft Comput*, 75:84-105. https://doi.org/10.1016/j.asoc.2018.10.044

Tripathi AK, Sharma K, Bala M, 2018. A novel clustering method using enhanced grey wolf optimizer and MapReduce. *Big Data Res*, 14:93-100. https://doi.org/10.1016/j.bdr.2018.05.002

Wu GH, Mallipeddi R, Suganthan PN, 2016. Problem Definitions and Evaluation Criteria for the CEC 2017 Competition and Special Session on Constrained Single Objective Real-Parameter Optimization. Technical Report, No. 201212, Nanyang Technological University, Singapore.

Zawbaa HM, Emary E, Grosan C, et al., 2018. Large-dimensionality small-instance set feature selection: a hybrid bio-inspired heuristic approach. *Swarm Evol Comput*, 42:29-42. https://doi.org/10.1016/j.swevo.2018.02.021

Zhang S, Zhou YQ, 2015. Grey wolf optimizer based on Powell local optimization method for clustering analysis. *Discr Dynam Nat Soc*, 2015:481360. https://doi.org/10.1155/2015/481360

Zhang XM, Kang Q, Cheng JF, et al., 2018. A novel hybrid algorithm based on biogeography-based optimization and grey wolf optimizer. *Appl Soft Comput*, 67:197-214. https://doi.org/10.1016/j.asoc.2018.02.049

Zhang XQ, Ming ZF, 2017. An optimized grey wolf optimizer based on a mutation operator and eliminating-reconstructing mechanism and its application. *Front Inform Technol Electron Eng*, 18(11):1705-1719. https://doi.org/10.1631/FITEE.1601555