

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**SOICT**

**BÁO CÁO TÌM HIỂU**  
**THUẬT TOÁN GREY WOLF OPTIMIZATION**

STT	Họ và tên	Mã số sinh viên
1	Vũ Thị Hợp	202417240
2	Lê Thu Ngân	202417257

Người hướng dẫn : Tạ Duy Lâm

# Mục lục

<b>1</b>	<b>Giới thiệu về phương pháp tối ưu Meta-heuristic</b>	<b>2</b>
<b>2</b>	<b>Grey Wolf Optimization</b>	<b>2</b>
2.1	Tìm hiểu chung về sự phân cấp của sói xám và hành vi săn mồi . . . . .	2
2.2	Mô hình toán học và thuật toán GWO . . . . .	2
2.2.1	Tìm kiếm và vây bắt con mồi . . . . .	3
2.2.2	Tấn công con mồi . . . . .	4
2.2.3	Mã giả GWO . . . . .	5
2.3	Độ phức tạp thuật toán . . . . .	6
2.3.1	Time complexity . . . . .	6
2.3.2	Bảng tóm tắt độ phức tạp . . . . .	6
2.4	Đánh giá thuật toán . . . . .	6
2.4.1	Đánh giá chất lượng thuật toán . . . . .	6
2.4.2	Phân tích khả năng tiếp cận (exploitation) . . . . .	8
2.4.3	Phân tích khả năng tìm kiếm (exploration) . . . . .	9
2.4.4	Khả năng tránh cực trị địa phương . . . . .	9
2.4.5	Phân tích sự hội tụ . . . . .	9
2.5	Kết luận . . . . .	10
<b>3</b>	<b>Các biến thể của GWO</b>	<b>11</b>
3.1	Biến thể I-GWO 2024 by Yihui Qiu, Xiaoxiao Yang & Shuixuan Chen . .	11
3.1.1	Mục đích của thuật toán IGWO . . . . .	11
3.1.2	Phương pháp thấu kính ngược . . . . .	11
3.1.3	Điều chỉnh biến điều khiển a phi tuyến . . . . .	12
3.1.4	Điều chỉnh cơ chế tìm kiếm . . . . .	13
3.1.5	Mã giả của thuật toán IGWO . . . . .	14
3.2	Biến thể I-GWO 2020 by Mohammad H. Nadimi-Shahraki, Shokooh Taghian, Seyedali Mirjalili . . . . .	14
3.2.1	Mục đích của thuật toán IGWO . . . . .	14
3.2.2	Khởi tạo . . . . .	14
3.2.3	Movement phase . . . . .	15
3.2.4	Lựa chọn và cập nhật . . . . .	15
3.2.5	Mã giả . . . . .	16
3.3	Multiple Objective Function . . . . .	16
3.3.1	Giới thiệu thuật toán Multiple Objective Function . . . . .	16
3.3.2	Kho lưu trữ . . . . .	17
3.3.3	Cơ chế lưới (Grid Mechanism) . . . . .	17
3.3.4	Cơ chế chọn thủ lĩnh . . . . .	18
3.3.5	Mã giả MOGWO . . . . .	19
<b>4</b>	<b>References</b>	<b>20</b>

# 1 Giới thiệu về phương pháp tối ưu Meta-heuristic

Phương pháp tối ưu Meta-heuristic đã trở nên rất phổ biến nhờ tính đơn giản, linh hoạt, cơ chế không cần đạo hàm và tránh cực trị cục bộ. Swarm Intelligence (SI) là một nhánh của phương pháp tối ưu Meta-heuristic. Nguồn cảm hứng của các kỹ thuật SI chủ yếu xuất phát từ các đàn vật nuôi tự nhiên, đàn chim, bầy thú và đàn cá. Một số kỹ thuật SI phổ biến nhất là ACO (Ant colony optimization), PSO (Particle swarm optimization), và Artificial bee colony (ABC).

Nhìn chung, một đặc điểm phổ biến của của Meta-heuristic là việc chia quá trình tìm kiếm thành 2 giai đoạn: khám phá (exploration) và tiếp cận (exploitation). Chức năng của hai quá trình này đóng vai trò quan trọng trong việc tìm kiếm lời giải tối ưu. GWO là một kỹ thuật tối ưu bầy đàn (SI) mới bởi Seyedali Mirjalili et al. Thuật toán được truyền cảm hứng bởi sự phân cấp xã hội và hành vi săn mồi của bầy sói xám.

## 2 Grey Wolf Optimization

### 2.1 Tìm hiểu chung về sự phân cấp của sói xám và hành vi săn mồi

Trong GWO, quần thể các giải pháp (sói) được chia thành bốn cấp bậc dựa trên độ thích nghi (fitness) của chúng, mô phỏng hệ thống phân cấp xã hội của bầy sói thật:

- **Alpha ( $\alpha$ ):** Là con sói đầu đàn, đại diện cho **giải pháp tốt nhất** tìm được cho đến hiện tại. Quyết định của Alpha là quan trọng nhất trong việc săn mồi.
- **Beta ( $\beta$ ):** Là con sói đứng thứ hai trong hệ thống phân cấp, hỗ trợ Alpha trong việc ra quyết định và có thể thay thế Alpha nếu Alpha chết hoặc không còn phù hợp. Beta đại diện cho **giải pháp tốt thứ hai**.
- **Delta ( $\delta$ ):** Là con sói đứng thứ ba, tuân theo Alpha và Beta nhưng có quyền chỉ huy các con sói Omega. Delta đại diện cho **giải pháp tốt thứ ba**.
- **Omega ( $\omega$ ):** Là những con sói còn lại trong bầy, tuân theo sự chỉ đạo của Alpha, Beta và Delta. Chúng đại diện cho các **giải pháp ứng viên khác**.

Trong thuật toán GWO, ba con sói tốt nhất (**Alpha, Beta, Delta**) được sử dụng để dẫn dắt quá trình tìm kiếm, trong khi các con sói Omega cập nhật vị trí của chúng dựa trên vị trí của ba con sói đầu đàn này.

Quá trình săn mồi của sói xám, theo Muro và các cộng sự bao gồm **3 giai đoạn chính**:

- Rình rập, truy đuổi và tiếp cận con mồi
- Bao vây con mồi khi chúng dừng lại hoặc yếu đi
- Tấn công con mồi

### 2.2 Mô hình toán học và thuật toán GWO

Quá trình săn mồi của sói xám bao gồm các bước sau:

### 2.2.1 Tìm kiếm và vây bắt con mồi

Ban đầu, các con sói (giải pháp) được khởi tạo ngẫu nhiên trong không gian tìm kiếm. Chúng di chuyển để tìm kiếm con mồi (giải pháp tối ưu). Quy trình này được mô tả bởi hai phương trình sau:

$$\vec{D} = \left| \vec{C} \cdot \vec{X}_p(t) - \vec{X}(t) \right| \quad (1)$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \quad (2)$$

Trong đó:

- $t$  là vòng lặp hiện tại.
- $\vec{X}_p$  là vectơ vị trí của con mồi (best candidate solution).
- $\vec{X}$  là vectơ vị trí của một con sói trong quần thể.
- $\vec{D}$  là khoảng cách giữa vị trí con mồi (giải pháp) và vị trí của con sói hiện tại.
- $\vec{A}, \vec{C}$  là vectơ hệ số, được tính toán như sau:

$$\vec{A} = 2 \cdot \vec{a} \cdot \vec{r}_1 - \vec{a} \quad (3)$$

$$\vec{C} = 2 \cdot \vec{r}_2 \quad (4)$$

Trong đó:

- $\vec{r}_1, \vec{r}_2$  là các vectơ ngẫu nhiên trong khoảng  $[0, 1]$ .
- $\vec{a}$  là vectơ có vai trò điều khiển, giảm tuyến tính từ  $[2, 0]$  theo số vòng lặp.

Vectơ  $\vec{a}$  có vai trò quan trọng trong thuật toán để cân bằng giữa khả năng tìm kiếm (*exploration*) và tấn công (*exploitation*). Cụ thể:

- **Giai đoạn đầu:** khi  $a \rightarrow 2$  dẫn đến  $|\vec{A}| > 1$ . Điều này có nghĩa trong lần cập nhật vị trí tiếp theo, các con sói có xu hướng di chuyển rời xa vị trí con mồi. Điều này tương ứng với giai đoạn tìm kiếm các vùng mới trong không gian.
- **Giai đoạn sau:** khi  $a \rightarrow 0$  dẫn đến  $|\vec{A}| < 1$ . Lúc này, vị trí của các con sói đang dần tiến về phía con mồi, tức là bước vào giai đoạn khai thác điểm tối ưu.

Vectơ  $\vec{C}$  có vai trò xê dịch nhỏ một cách ngẫu nhiên vị trí của con mồi giúp tránh lỗi cực trị địa phương, giúp sói có thể thoát ra và tìm kiếm giải pháp tối ưu hơn. Kể cả ở vòng lặp hiện tại hay những vòng lặp cuối.

Mỗi thành phần trong một vectơ đại diện cho một biến trong không gian tìm kiếm. Nếu số chiều là 1, vị trí con sói nằm trên trục số; nếu số chiều là 2, vị trí là cặp tọa độ  $(x, y)$ ; nếu số chiều là 3, vị trí là  $(x, y, z)$ .

### 2.2.2 Tấn công con mồi

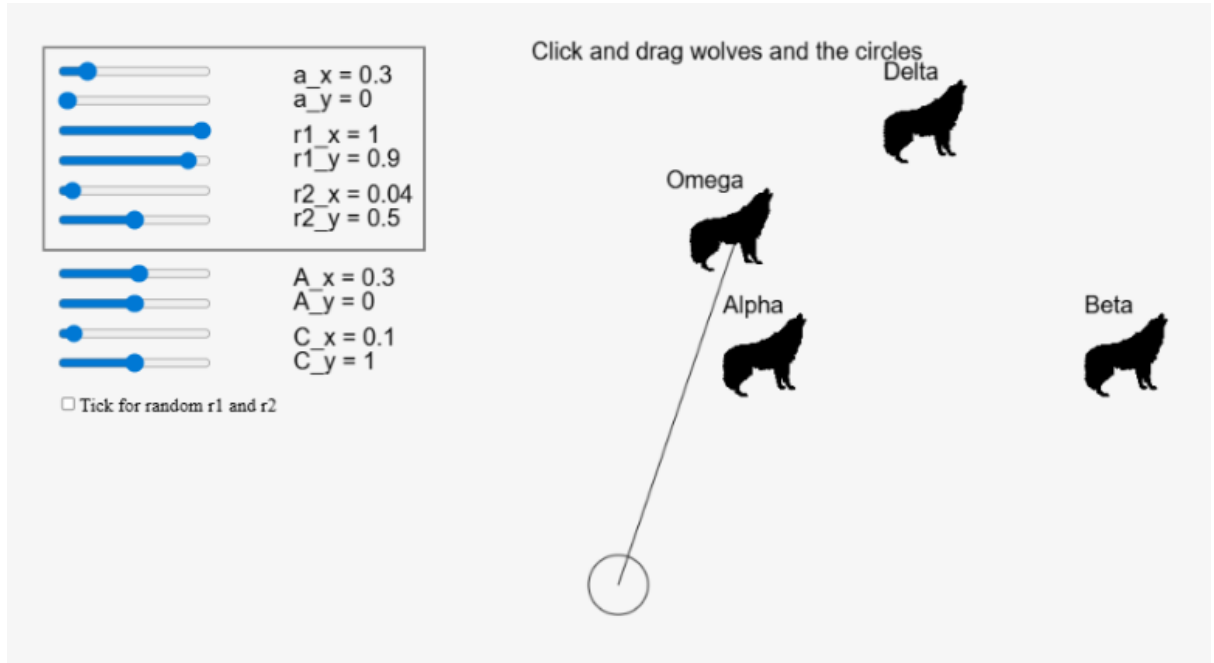
Sói xám có khả năng nhận biết vị trí của con mồi và bao vây chúng. Tuy nhiên, trong không gian tìm kiếm giải pháp tối ưu, chúng ta không biết vị trí của điểm tối ưu (con mồi). Vì vậy ta giả sử rằng vị trí  $X_\alpha$  là giải pháp ứng viên tốt nhất, tiếp đến là  $X_\beta$ ,  $X_\delta$  và cập nhật các vị trí của các con sói còn lại theo vị trí của các giải pháp tốt ở trên. Mô hình toán học của chúng được thể hiện như sau:

$$\vec{D}_\alpha = |\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}|, \quad \vec{D}_\beta = |\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}|, \quad \vec{D}_\delta = |\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}| \quad (5)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot (\vec{D}_\alpha), \quad \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot (\vec{D}_\beta), \quad \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot (\vec{D}_\delta) \quad (6)$$

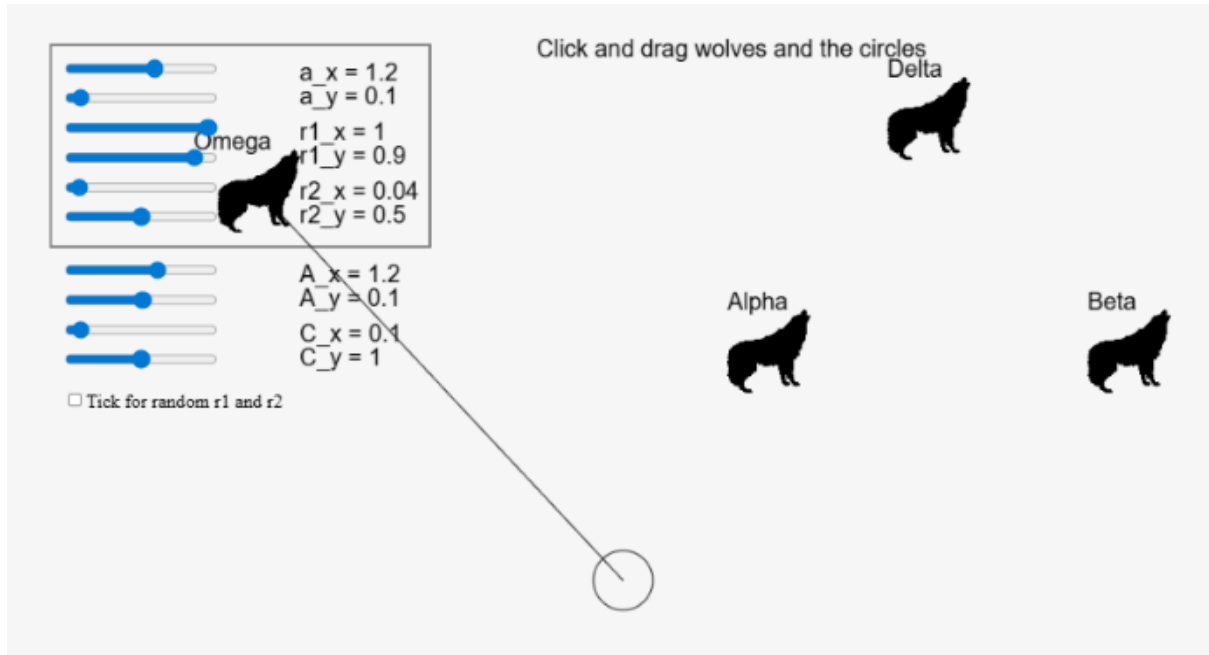
$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (7)$$

Hình ảnh mô phỏng vị trí cập nhật của sói Omega theo vị trí của 3 con sói Alpha, Beta và Delta trong không gian 2D.



Như đã giải thích vai trò của vectơ hệ số A ở trên. Do  $A_x$  và  $A_y$  đều  $< 1$ . Nên vị trí của Omega có xu hướng gần với tâm của 3 con sói Alpha, Beta và Delta.

Trường hợp khi  $A_x > 1$ , ở trục  $X_\omega$  có xu hướng rời xa con mồi, được mô tả như hình dưới đây.



### 2.2.3 Mã giả GWO

- 
1. Initialize the grey wolf population  $X_i$  ( $i = 1, 2, \dots, n$ )
  2. Initialize  $a$ ,  $A$ , and  $C$
  3. Initialize the fitness of each search agent
  4.  $X_\alpha$  = the best search agent
  5.  $X_\beta$  = the second best search agent
  6.  $X_\delta$  = the third best search agent
  7. **while** ( $t < \text{Max number of iterations}$ )
  8.     for each search agent
  9.         Update the position of the current search agent
  10.     end for
  11.     Update  $a$ ,  $A$ , and  $C$
  12.     Calculate the fitness of all search agents
  13.     Updating  $X_\alpha$ ,  $X_\beta$  and  $X_\delta$
  14.      $t = t + 1$
  15. **end while**
  16. Return  $X_\alpha$
-

## 2.3 Độ phức tạp thuật toán

### 2.3.1 Time complexity

Độ phức tạp tổng quát Time complexity:

$$O(\text{MaxT} \times n \times C_f)$$

Trong đó:

- **MaxT**: Số vòng lặp tối đa (iterations)
- **n**: Kích thước quần thể (số wolves)
- **$C_f$** : Độ phức tạp của hàm mục tiêu (fitness function)

### 2.3.2 Bảng tóm tắt độ phức tạp

Thành phần	Time Complexity	Space Complexity
Initialization	$O(n \times D)$	$O(n \times D)$
Fitness Evaluation	$O(n \times C_f)$	$O(1)$
Position Update	$O(n \times D)$	$O(1)$
Per Iteration	$O(n \times C_f)$	$O(n \times D)$
Total	$O(\text{MaxT} \times n \times C_f)$	$O(n \times D)$

## 2.4 Đánh giá thuật toán

### 2.4.1 Đánh giá chất lượng thuật toán

Thuật toán GWO được đánh giá trên 29 hàm tiêu chuẩn và so sánh kết quả với các thuật toán metaheuristic khác. Những hàm chuẩn được liệt kê trong các bảng 1, bảng 2, bảng 3, bảng 4 với Dim là chiều của hàm, Range là miền giới hạn của không gian tìm kiếm và  $f_{\min}$  là kết quả tối ưu.

Nhìn chung, hàm tiêu chuẩn được chia thành 4 nhóm: hàm đơn cực trị (unimodal), hàm đa cực trị (multimodal), hàm đa cực trị với số chiều cố định (fixed-dimension multimodal), và hàm hợp (composite functions).

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  x_i  + \prod_{i=1}^n  x_i $	30	[-10,10]	0
$f_3(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	30	[-100,100]	0
$f_4(x) = \max_i \{ x_i , 1 \leq i \leq n\}$	30	[-100,100]	0
$f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i)^2 + (x_i - 1)^2]$	30	[-30,30]	0
$f_6(x) = \sum_{i=1}^n ([x_i + 0.5])^2$	30	[-100,100]	0
$f_7(x) = \sum_{i=1}^n ix_i^4 + random[0,1]$	30	[-1.28,1.28]	0

Table 2. Multimodal benchmark functions

Function	Dim	Range	$f_{min}$
$F_8(x) = \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	30	[-500,500]	-418.9829×5
$F_9(x) = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	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
$F_{12}(x) = \frac{\pi}{n} \{10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^n u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$	30	[-50,50]	0
$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$			0
$F_{13}(x) = 0.1 \{ \sin^2(3\pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \} + \sum_{i=1}^n u(x_i, 5, 100, 4)$	30	[-50,50]	0
$F_{14}(x) = -\sum_{i=1}^n \sin(x_i) \cdot \left( \sin\left(\frac{ix_i}{\pi}\right) \right)^{2m}, \quad m = 10$	30	[0,π]	-4.687
$F_{15}(x) = \left[ e^{-\sum_{i=1}^n (x_i/\beta)^{2m}} - 2e^{-\sum_{i=1}^n x_i^2} \right] \cdot \prod_{i=1}^n \cos^2 x_i, \quad m = 5$	30	[-20,20]	-1
$F_{16}(x) = \{ [\sum_{i=1}^n \sin^2(x_i)] - \exp(-\sum_{i=1}^n x_i^2) \} \cdot \exp[-\sum_{i=1}^n \sin^2 \sqrt{ x_i }]$	30	[-10,10]	-1

Table 3. Fixed-dimension multimodal benchmark functions

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



Bảng so sánh kết quả thống kê với các thuật toán khác được báo cáo trong bảng 5 đến bảng 8.

Table 5. Results of unimodal benchmark functions

F	GWO		PSO		GSA		DE		FEP	
	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>
F1	6.59E-28	6.34E-05	0.000136	0.000202	2.53E-16	9.67E-17	8.2E-14	5.9E-14	0.00057	0.00013
F2	7.18E-17	0.029014	0.042144	0.045421	0.055655	0.194074	1.5E-09	9.9E-10	0.0081	0.00077
F3	3.29E-06	79.14958	70.12562	22.11924	896.5347	318.9559	6.8E-11	7.4E-11	0.016	0.014
F4	5.61E-07	1.315088	1.086481	0.317039	7.35487	1.741452	0	0	0.3	0.5
F5	26.81258	69.90499	96.71832	60.11559	67.54309	62.22534	0	0	5.06	5.87
F6	0.816579	0.000126	0.000102	8.28E-05	2.5E-16	1.74E-16	0	0	0	0
F7	0.002213	0.100286	0.122854	0.044957	0.089441	0.04339	0.00463	0.0012	0.1415	0.3522

Table 6. Results of multimodal benchmark functions

F	GWO		PSO		GSA		DE		FEP	
	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>
F8	-6123.1	-4087.44	-4841.29	1152.814	-2821.07	493.0375	-11080.1	574.7	-12554.5	52.6
F9	0.310521	47.35612	46.70423	11.62938	25.96841	7.470068	69.2	38.8	0.046	0.012
F10	1.06E-13	0.077835	0.276015	0.50901	0.062087	0.23628	9.7E-08	4.2E-08	0.018	0.0021
F11	0.004485	0.006659	0.009215	0.007724	27.70154	5.040343	0	0	0.016	0.022
F12	0.053438	0.020734	0.006917	0.026301	1.799617	0.95114	7.9E-15	8E-15	9.2E-06	3.6E-06
F13	0.654464	0.004474	0.006675	0.008907	8.899084	7.126241	5.1E-14	4.8E-14	0.00016	0.000073

Table 7. Results of fixed-dimension multimodal benchmark functions

F	GWO		PSO		GSA		DE		FEP	
	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>
F14	4.042493	4.252799	3.627168	2.560828	5.859838	3.831299	0.998004	3.3E-16	1.22	0.56
F15	0.000337	0.000625	0.000577	0.000222	0.003673	0.001647	4.5E-14	0.00033	0.0005	0.00032
F16	-1.03163	-1.03163	-1.03163	6.25E-16	-1.03163	4.88E-16	-1.03163	3.1E-13	-1.03	4.9E-07
F17	0.397889	0.397887	0.397887	0	0.397887	0	0.397887	9.9E-09	0.398	1.5E-07
F18	3.000028	3	3	1.33E-15	3	4.17E-15	3	2E-15	3.02	0.11
F19	-3.86263	-3.86278	-3.86278	2.58E-15	-3.86278	2.29E-15	N/A	N/A	-3.86	0.000014
F20	-3.28654	-3.25056	-3.26634	0.060516	-3.31778	0.023081	N/A	N/A	-3.27	0.059
F21	-10.1514	-9.14015	-6.8651	3.019644	-5.95512	3.737079	-10.1532	0.0000025	-5.52	1.59
F22	-10.4015	-8.58441	-8.45653	3.087094	-9.68447	2.014088	-10.4029	3.9E-07	-5.53	2.12
F23	-10.5343	-8.55899	-9.95291	1.782786	-10.5364	2.6E-15	-10.5364	1.9E-07	-6.57	3.14

Table 8. Results of composite benchmark functions

F	GWO		PSO		GSA		DE		CMA-ES	
	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>	<i>ave</i>	<i>std</i>
F24	43.83544	69.86146	100	81.65	6.63E-17	2.78E-17	6.75E-02	1.11E-01	100	188.56
F25	91.80086	95.5518	155.91	13.176	200.6202	67.72087	28.759	8.6277	161.99	151
F26	61.43776	68.68816	172.03	32.769	180	91.89366	144.41	19.401	214.06	74.181
F27	123.1235	163.9937	314.3	20.066	170	82.32726	324.86	14.784	616.4	671.92
F28	102.1429	81.25536	83.45	101.11	200	47.14045	10.789	2.604	358.3	168.26
F29	43.14261	84.48573	861.42	125.81	142.0906	88.87141	490.94	39.461	900.26	8.32E-02

## 2.4.2 Phân tích khả năng tiếp cận (exploitation)

Theo bảng kết quả 5, thuật toán GWO đưa ra kết quả tối ưu hơn, đặc biệt ở các hàm F1, F2 và F7. Và bởi các hàm đơn cực trị phù hợp cho đánh giá mức độ tiếp cận hay khai thác kết quả tối ưu. Vì vậy mà kết quả này cho thấy khả năng khai thác tối ưu của thuật toán GWO có hiệu suất vượt trội.

### 2.4.3 Phân tích khả năng tìm kiếm (exploration)

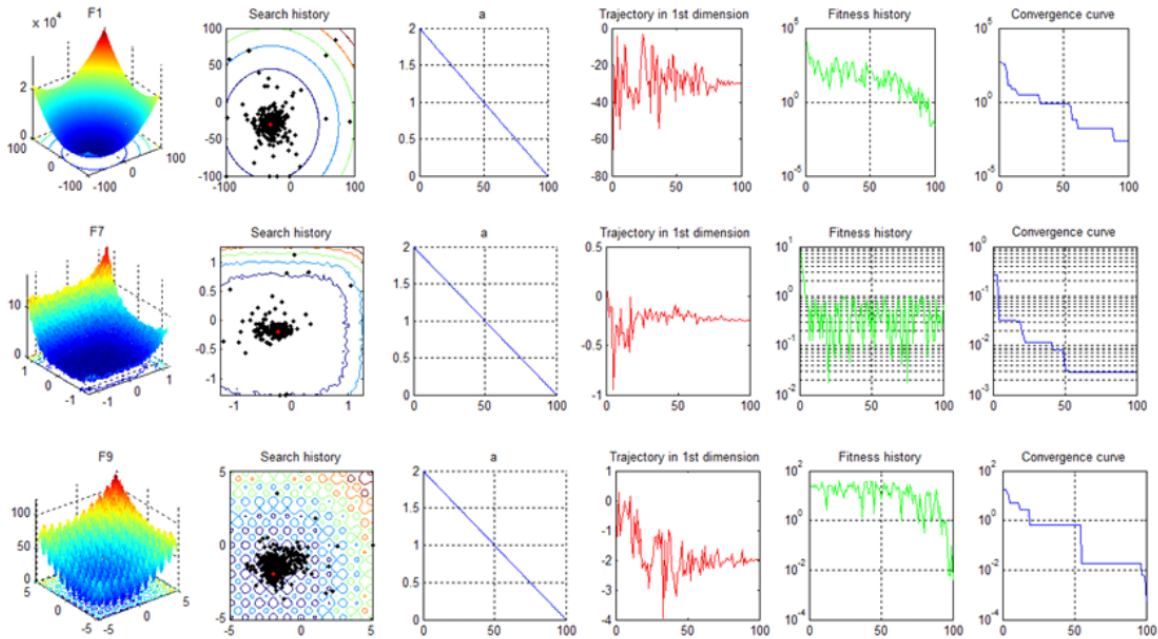
So với các hàm có nhiều cực trị địa phương các hàm này phù hợp cho việc đánh giá mức độ tìm kiếm, khám phá các vùng mới. Theo kết quả bảng 6,7 cũng cho thấy hiệu suất tốt cho các hàm đa cực trị. Thuật toán GWO hầu như có kết quả tốt hơn so với PSO và GSA. So với thuật toán DE và FEP, nó có kết quả khá tương đương và thỉnh thoảng vượt trội hơn. Vì vậy về mặt tìm kiếm, thuật toán GWO cũng có hiệu suất tốt.

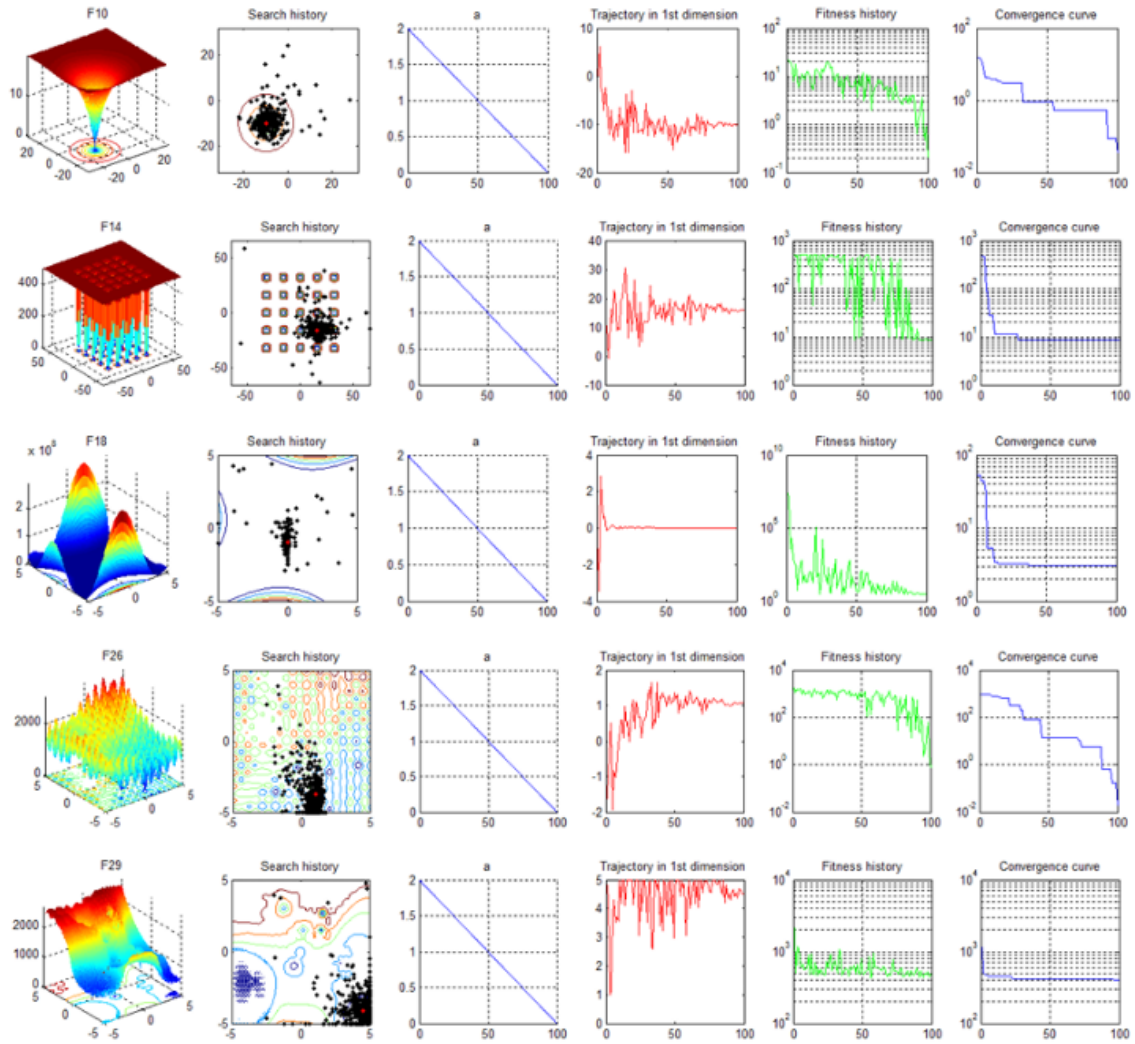
### 2.4.4 Khả năng tránh cực trị địa phương

Với các hàm hợp thường là các thách thức so với các thuật toán để thử nghiệm đồng thời khả năng tìm kiếm và khai thác kết quả tối ưu. Theo Bảng 8, ta có thể thấy kết quả của thuật toán GWO vượt trội ở phần đầu và cũng có kết quả khá tương đối ở phần còn lại. Điều này cho thấy khả năng cân bằng giữa tìm kiếm và khai thác của thuật toán GWO nhờ vào khả năng thích ứng của  $A$  như đã đề cập ở trên:  $|A| \geq 1$  ở giai đoạn đầu và  $|A| < 1$  ở giai đoạn sau.

### 2.4.5 Phân tích sự hội tụ

Trong những giai đoạn đầu vòng lặp, để tránh rơi vào vòng lặp và kẹt ở cực trị địa phương, những ứng viên tìm kiếm (các con sói) thường có những thay đổi đột biến của chuyển động ở giai đoạn đầu của quá trình tìm kiếm tối ưu. Sau đó, những thay đổi này sẽ giảm dần để nhấn mạnh việc tiếp cận vị trí tối ưu. Vì vậy nên ở những vòng lặp đầu, sự hội tụ thường giảm nhẹ vì chúng đang ở quá trình tìm kiếm. Phân tích sự hội tụ được minh họa ở bảng 11. Trong đó cột thứ 4 là diễn biến đường đi của thành phần đầu tiên (hay con sói đầu tiên) của chiều thứ nhất trong quá trình tìm kiếm. Ta thấy ở những vòng đầu, vị trí của nó có biến động lớn và giảm dần ở những vòng lặp cuối.





## 2.5 Kết luận

Grey Wolf Optimization là một thuật toán tối ưu hoá bầy đàn (SI) lấy cảm hứng từ loài sói xám. Nó mô tả cấu trúc xã hội và hành vi săn mồi của bầy sói. Kết quả thử nghiệm trên 29 hàm tiêu chuẩn cho thấy hiệu suất toàn diện của thuật toán, sự cân bằng giữa tìm kiếm và khai thác tối ưu, cũng như khả năng tránh kẹt ở các cực trị địa phương và sự hội tụ của thuật toán.

### 3 Các biến thể của GWO

#### 3.1 Biến thể I-GWO 2024 by Yihui Qiu, Xiaoxiao Yang & Shuixuan Chen

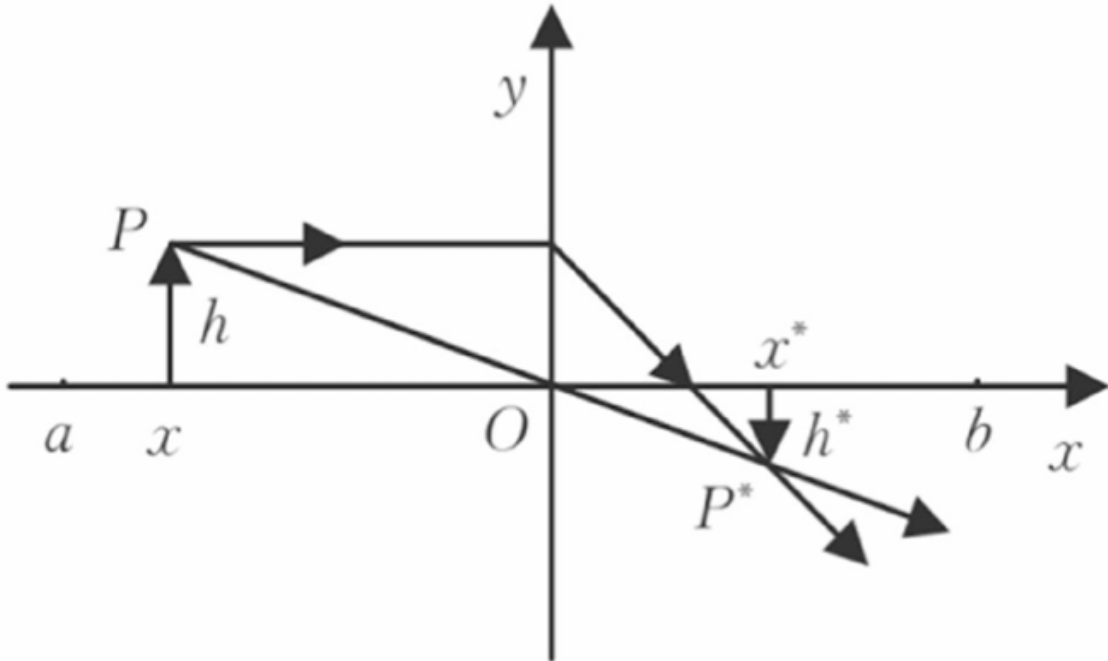
##### 3.1.1 Mục đích của thuật toán IGWO

Nhằm tăng hiệu suất cho thuật toán truyền thống GWO, ba cải tiến được đề xuất trong thuật toán IGWO đó là tối ưu số lượng ứng viên tìm kiếm, điều chỉnh biến điều khiển  $a$  phi tuyến, và cơ chế tìm kiếm (exploration). Những cải tiến này nhằm tăng khả năng tìm kiếm những vùng mới, tăng khả năng tìm kiếm tối ưu, giảm quần thể giúp tối ưu độ phức tạp thuật toán.

##### 3.1.2 Phương pháp thấu kính ngược

Ở thuật toán gốc, các vòng lặp sau, lượng sói xám tập trung rất nhiều ở vùng lân cận điểm tối ưu. Điều này có thể dẫn đến việc khó nhả ra khỏi vùng cực tiểu địa phương dẫn đến hiện tượng kết thúc thuật toán sớm và giảm độ chính xác cho việc tìm kiếm điểm tối ưu. Thấu kính ngược, tức là nhìn vào điểm tối ưu cả qua thấu kính và các vị trí  $x$  tương đương với điểm  $x^*$  ngược qua  $O$ , giúp tăng khả năng tìm kiếm và tránh bị bế. Khi đó, ta có được quy tắc ngược ảnh như sau:

$$\frac{(a+b)/2 - x}{x^* - (a+b)/2} = \frac{h}{h^*} \quad (8)$$



Sơ đồ nguyên lý học ngược (Reverse learning) tạo ảnh thấu kính.

Trong đó,  $h$  là khoảng cách từ vị trí  $x$  đến điểm tối ưu, với 2 biến  $a, b$ .

Mở rộng trong không gian  $d$  chiều ta có:

$$X_i^{d*} = \frac{X_{\min}^d + X_{\max}^d}{2} + \frac{X_{\min}^d + X_{\max}^d}{2k} - \frac{X_i^d}{k} \quad (9)$$

Với  $k = \frac{h}{h^*}$

Khi  $k$  nhỏ dần thì phạm vi điểm tối ưu “ngược” dần ra xa. Điều này làm tăng khả năng tìm kiếm vùng rộng. Ngược lại,  $k$  lớn dần thì phạm vi điểm tối ưu ngược nhỏ dần lại, khiến cho việc tinh chỉnh kết quả tối ưu tốt. Để mô tả đúng đặc tính của thuật GWO, đó là tìm kiếm rộng ở giai đoạn đầu vòng lặp và tinh chỉnh tìm kiếm ở vòng lặp cuối, biểu diễn toán học cho  $k$  được thể hiện như sau:

$$k(t) = \left(1 + \left(\frac{t}{T_{\max}}\right)^{1/2}\right)^8 \quad (10)$$

Trong đó:

- $t$ : Vòng lặp hiện tại
- $T_{\max}$ : Số vòng lặp tối đa

### 3.1.3 Điều chỉnh biến điều khiển $a$ phi tuyến

Giá trị của  $a$  đóng vai trò quan trọng cho giá trị  $A$ , để xác định việc tìm kiếm hay khai thác vị trí tối ưu. Tuy nhiên, việc giảm tuyến tính  $a$  không hoàn toàn phản ánh đúng thực tế quá trình tìm kiếm phi tuyến, dẫn đến thuật toán có thể rơi vào vòng cực tiểu địa phương. Để giải quyết vấn đề này thì thuật toán IGWO đã đề xuất một hàm phi tuyến cho  $a$ , được thể hiện như sau:

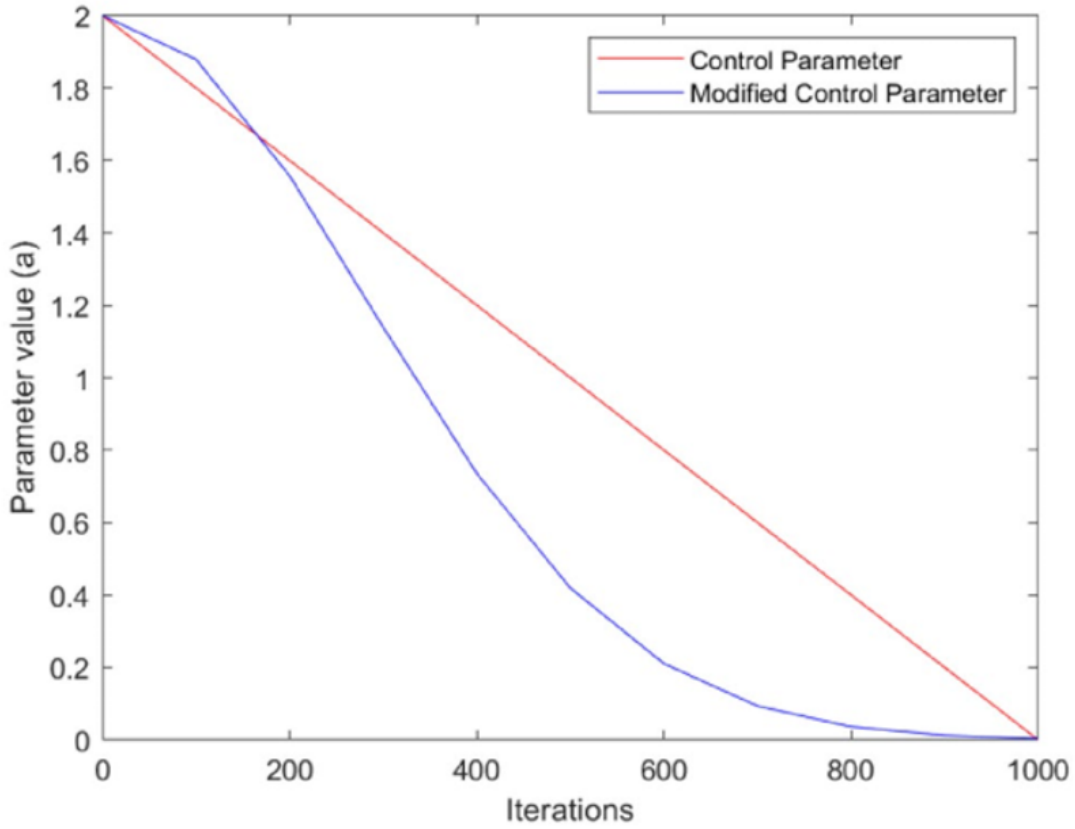
$$a = (a_{\text{initial}} - a_{\text{end}}) * \exp\left(-\frac{t^2}{(K * T_{\max})^2}\right) + a_{\text{end}} \quad (11)$$

Trong đó:

- $t$ : Vòng lặp hiện tại
- $T_{\max}$ : Số vòng lặp tối đa
- $K$ : Hệ số điều chế, cố định bằng 0.3
- $a_{\text{initial}}$  và  $a_{\text{end}}$ : Hệ số khởi tạo và kết thúc của  $a$ . Thiết lập  $a_{\text{initial}} = 2$ ,  $a_{\text{end}} = 0$ .

Dựa trên bảng giá trị so sánh giữa  $a$  tuyến tính và  $a$  phi tuyến, ta có thể thấy:

- Đối với  $a$  tuyến tính, thuật toán dành 50% số vòng lặp cho quá trình tìm kiếm (exploration) và 50% còn lại cho quá trình khai thác (exploitation).
- Ngược lại, với  $a$  phi tuyến, chỉ khoảng 25% số lượng vòng lặp ban đầu được dùng để tìm kiếm, phần lớn thời gian còn lại tập trung vào việc khai thác điểm tối ưu. Chiến lược này nhằm tăng cường khả năng khai thác cục bộ, đồng thời giảm thiểu rủi ro bỏ qua giá trị tối ưu thực sự.



Đường cong giảm tuyến tính và phi tuyến của tham số a.

### 3.1.4 Điều chỉnh cơ chế tìm kiếm

Theo thuật toán gốc, ta giả định 3 ứng viên tốt nhất Alpha, Beta, Delta và các ứng viên tiếp theo được cập nhật theo tâm của 3 vị trí trên. Tuy nhiên, nếu chúng bị nghẽn ở điểm hay định hướng bị lệch xa so với điểm tối ưu thực tế, sẽ rất mất thời gian hoặc hoạt động không hiệu quả. Vì vậy thuật toán IGWO đã đề xuất một cơ chế tìm kiếm mới, lấy cảm hứng từ thuật toán TSA và PSO.

$$X(t+1) = (w_1 * X_1 + w_2 * X_2 + (b_1 * r_3 * (pbest_i(t) - X(t)) + b_2 * r_4 * (X_1 - X(t)))) / (randn() + 2) \quad (12)$$

Trong đó:

- $X_1, X_2$ : Là vị trí con mồi được xác định từ vị trí của Alpha và Beta.
- $b_1, b_2$ : Là độ tin cậy vào hệ số trí nhớ của cá thể hay của loài. Thiết lập  $b_1 = b_2 = 0.5$ .
- $r_3, r_4$ : Là các số ngẫu nhiên trong khoảng  $[0, 1]$ .
- $pbest_i$ : Là vị trí tốt nhất của cá thể.
- $w_1, w_2$ : Cho biết mức độ ảnh hưởng của Alpha, Beta.

$$w_1 = \frac{f_\alpha}{f_\alpha + f_\beta} \quad w_2 = \frac{f_\beta}{f_\alpha + f_\beta} \quad (13)$$

Trong đó  $f_\alpha, f_\beta$  là giá trị thích nghi (fitness) của Alpha, Beta.

### 3.1.5 Mã giả của thuật toán IGWO

---

<b>Input:</b>	Population size is $N$ ; maximum number of iterations is $T$ ; current iteration $t = 1$ ; population dimension is $d$
<b>Output:</b>	Optimal solution $X_\alpha$

---

1. Use equation (9) to initialize the gray wolf population  $X_i$  ( $i = 1, 2, \dots, n$ )
2. Calculate the fitness value of each individual gray wolf;
3. Obtain the current top three gray wolf individuals in terms of fitness and define them as  $X_\alpha, X_\beta$  and  $X_\delta$  respectively;
4. **while** ( $t < T$ )
5.     for each gray wolf individual use equation (12) to update the position;
6.     update the control parameter  $a$  using equation (11) and update the associated parameter  $A$ ;
7.     updating the associated parameter  $C$
8.     calculating the fitness of each individual gray wolf;
9.     Updating  $X_\alpha, X_\beta$  and  $X_\delta$  ;
10.      $t = t + 1$
11. **end while**
12. Return  $X_\alpha$

---

## 3.2 Biến thể I-GWO 2020 by Mohammad H. Nadimi-Shahraki, Shokooch Taghian, Seyedali Mirjalili

### 3.2.1 Mục đích của thuật toán IGWO

Tương tự, thuật toán IGWO cũng nhằm cải tiến để tránh lỗi tắc nghẽn ở cực trị địa phương. Sự cải tiến này bao gồm chiến thuật mới cho sự lựa chọn và cập nhật ở các bước, được mô tả ở figure sau. Vì vậy, thuật toán I-GWO bao gồm ba giai đoạn: Khởi tạo (Initilization), Di chuyển (Move) và lựa chọn và cập nhật cho các bước tiếp theo.

### 3.2.2 Khởi tạo

Khởi tạo vị trí  $N$  sói theo phân bố ngẫu nhiên từ  $[l_j, u_j]$ :

$$X_{ij} = l_j + rand_j[0, 1] \times (u_j - l_j), \quad i \in [1, N], j \in [1, D] \quad (14)$$

Trong đó:

- $i$  là vị trí sói thứ  $i$ ,  $j$  là ở chiều thứ  $j$  trong không gian  $D$  chiều.
- Tất cả quần thể sói được lưu trữ trong ma trận Pop,  $N$  hàng và  $D$  cột.
- Fitness value của  $X_i(t)$  được tính toán bởi hàm  $f(X_i(t))$ .

### 3.2.3 Movement phase

Ngoài việc, di săn mỗi theo bầy đàn, sói xám cũng có săn mỗi theo từng cá thể. Đây là nguồn cảm hứng cho thuật toán IGWO, được gọi là phương pháp tìm kiếm DLH (Dimension learning-based hunting).

Phương pháp tìm kiếm này chỉ ra rằng, mỗi chiều của một vị trí mới của sói  $X_i$  được tính bởi Phương trình (17). Trong đó, mỗi cá thể bị ảnh hưởng bởi vị trí của những con sói hàng xóm của nó và bởi một con sói ngẫu nhiên trong bầy, giúp tăng sự đa dạng trong quá trình tìm kiếm.

Bên cạnh đó, chiến lược DLH cũng tạo ra một ứng viên khác cho vị trí mới của sói  $X_i(t)$  là  $X_{i-DLH}(t+1)$ . Điều này được thực hiện bằng cách: đầu tiên là tính khoảng cách Euclid giữa vị trí hiện tại và vị trí của  $X_{i-GWO}(t+1)$  (vị trí cho vòng lặp kế tiếp theo thuật toán GWO truyền thống, Phương trình (15)).

$$R_i(t) = \|X_i(t) - X_{i-GWO}(t+1)\| \quad (15)$$

Những con sói hàng xóm  $X_j$  là những con có khoảng cách Euclid so với  $X_i \leq R_i$ . Số lượng các con sói hàng xóm  $N_i(t)$  của  $X_i(t)$  được xây dựng ở Phương trình (16) như sau:

$$N_i(t) = \{X_j(t) \mid D_i(X_i(t), X_j(t)) \leq R_i(t), X_j(t) \in Pop\} \quad (16)$$

Sau khi xây dựng xong tập các con sói hàng xóm, vị trí của vòng tiếp theo của sói  $X_i(t)$  được cập nhật như sau:

$$X_{i-DLH,d}(t+1) = X_{i,d}(t) + rand \times (X_{n,d}(t) - X_{r,d}(t)) \quad (17)$$

Trong đó:

- $X_{n,d}(t)$ : là 1 con sói trong tập hàng xóm.
- $X_{r,d}(t)$ : là 1 con sói ngẫu nhiên trong bầy.

### 3.2.4 Lựa chọn và cập nhật

Mỗi ứng viên sẽ lựa chọn bước tiếp theo bằng cách so sánh giá trị tốt (fitness value) của  $X_{i-DLH}(t+1)$  hoặc  $X_{i-GWO}(t+1)$  theo Phương trình (18):

$$X_i(t+1) = \begin{cases} X_{i-GWO}(t+1), & \text{if } f(X_{i-GWO}) < f(X_{i-DLH}) \\ X_{i-DLH}(t+1) & \text{otherwise} \end{cases} \quad (18)$$

Sau đó, cập nhật vị trí của  $X_i(t+1)$  khi  $f(X_i(t+1)) < f(X_i(t))$ , ngược lại giữ nguyên.



### 3.2.5 Mã giả

---

**Input:**  $N, D, Maxiter$

**Output:** The global optimum

---

```

1 :   Begin
2 :   Initializing (Randomly distributing  $N$  wolves in the search space and
      calculating their fitness).
3 :   For  $iter = 2$  to  $Maxiter$ 
4 :       Find  $X_\alpha, X_\beta$ , and  $X_\delta$ .
5 :       For  $i = 1$  to  $N$ 
6 :           Computing  $X_{i1}, X_{i2}, X_{i3}$  by using Eq. (7).
7 :           Computing  $X_{i-GWO}(t+1)$  by using Eq. (8).
8 :           Calculating  $R_i(t)$  by Eq. (10).
9 :           Constructing neighborhood  $X_i(t)$  with radius  $R_i$  by Eq. (11).
10 :        For  $d = 1$  to  $D$ 
11 :             $X_{i-DLH,d}(t+1) = X_{i,d}(t) + rand \times (X_{n,d}(t) - X_{r,d}(t))$ 
12 :        End for
13 :        Selecting best  $(X_{i-GWO}(t+1), X_{i-DLH}(t+1))$ .
14 :        Updating Pop.
15 :    End for
16 : End for
17 : Return the global optimum.
18 : End

```

---

## 3.3 Multiple Objective Function

### 3.3.1 Giới thiệu thuật toán Multiple Objective Function

Theo thuật toán gốc GWO, dùng để tìm kiếm điểm tối ưu một hàm  $f(x)$ . Thì thuật toán MO GWO dùng để tối ưu một tập hợp các hàm. Ta có thể xây dựng một bài toán tối ưu hoá như sau:

$$\text{Maximize : } F(\vec{x}) = f_1(\vec{x}), f_2(\vec{x}), \dots, f_o(\vec{x}) \quad (19)$$

$$\text{Subject to : } g_i(\vec{x}) \geq 0, \quad i = 1, 2, \dots, m \quad (20)$$

$$h_i(\vec{x}) = 0, \quad i = 1, 2, \dots, p \quad (21)$$

$$L_i \leq x_i \leq U_i, \quad i = 1, 2, \dots, n \quad (22)$$

$F(x)$  là các hàm tối ưu.

Với inequality constraints (ràng buộc bất đẳng thức), và equality constraints (ràng buộc đẳng thức). Chúng có thể coi là điều kiện để việc tối ưu hoá trở nên có ý nghĩa. Trong đó:

- $L_i, U_i$ : Là các biên (giới hạn dưới và giới hạn trên).
- $n$ : Là số lượng biến (hay không gian  $D$  chiều).

Trong bài toán 1 chiều GWO gốc, ta có thể dễ dàng so sánh kết là tốt hay không. Nếu trong bài toán tối đa, kết quả  $x$  là tốt hơn  $y$  nếu  $f(x) > f(y)$ . Tuy nhiên, trong bài toán nhiều hàm cần tối đa, việc so sánh trở nên phức tạp hơn, và vì thế phương pháp dùng để so sánh được đề xuất bởi Francis Ysidro (Edgeworth, 1881) và sau đó được mở rộng bởi Vilfredo Pareto (Pareto, 1964).

Để giải quyết nhiều hàm, cách thành phần mới được tích hợp thêm vào GWO đó là: Kho lưu trữ và cơ chế chọn thủ lĩnh.

### 3.3.2 Kho lưu trữ

Kho lưu trữ (Archive): Là một đơn vị lưu trữ, để lưu trữ và truy xuất các giải pháp tối ưu Pareto.

Bản chất của kho lưu trữ:

- Kho lưu trữ có kích thước cố định.
- Chịu trách nhiệm duy trì các giải pháp không bị thống trị.

Quy tắc thống trị để cập nhật kho lưu trữ:

- Nếu giải pháp mới bị thống trị bởi các giải pháp đã có trong kho, thì giải pháp mới này không được cập nhật vào trong kho. Ngược lại, thì chúng được cập nhật, và giải pháp bị thống trị sẽ bị loại bỏ.
- Nếu nó không thống trị lẫn nhau, giải pháp mới được thêm vào kho.

### 3.3.3 Cơ chế lưới (Grid Mechanism)

Cơ chế lưới nhằm duy trì sự đa dạng: Kho lưu trữ có số lượng thành viên tối đa. Cơ chế lưới được tích hợp vào trong GWO nhằm cải thiện các giải pháp không bị thống trị trong kho.

Cơ chế lưới hay chính là không gian tìm kiếm chia không gian mục tiêu thành các vùng. Một số vùng sẽ có rất nhiều kết quả được lưu trữ ở kho, một số vùng khác thì lại thưa.

Việc chia không gian mục tiêu và quản lý các vùng này là cần thiết khi kho lưu trữ đạt đến kích thước tối đa:

Khi kho đầy thì cơ chế này sẽ được kích hoạt xóa 1 giải pháp ở vùng tập trung đông và thêm giải pháp mới vào vùng thưa trong lưới để cải thiện sự đa dạng tối ưu Pareto.

### 3.3.4 Cơ chế chọn thủ lĩnh

Trong thuật toán GWO gốc, ta chọn  $X_\alpha, X_\beta, X_\delta$  lần lượt là giải pháp tốt nhất. Tuy nhiên, trong bài toán giải quyết nhiều hàm, do không có 1 giải pháp “tốt nhất” duy nhất, nên  $X_\alpha, X_\beta, X_\delta$  được chọn từ kho lưu trữ. Việc chia không gian mục tiêu thành vùng đề cập ở trên là nền tảng cho Cơ chế Chọn Thủ Lĩnh.

- + Ưu tiên khu vực ít đông đúc để chọn  $X_\alpha, X_\beta, X_\delta$ . Điều này nhằm đảm bảo các sói được dẫn đến khám phá các vùng mới, giúp đa dạng kết quả và tránh các sói bị tắc nghẽn một vùng, dẫn đến sự hội tụ sớm.
- + Phương pháp Chọn: Việc lựa chọn được thực hiện bằng phương pháp bánh xe roulette với xác suất cho mỗi vùng như sau:

$$P_i = \frac{c}{N_i} \quad (23)$$

Trong đó:  $c$  là hằng số  $> 1$ ,  $N_i$  là số lượng kết quả tối ưu Pareto ở vùng thứ  $i$ . Có thể thấy xác suất tăng lên khi lượng kết quả tối ưu ở đó nhỏ, và ngược lại.

Các trường hợp cụ thể:

- Nếu số giải pháp ở đó  $\geq 3$ , thì ta chọn ngẫu nhiên Alpha, Beta, Delta.
- Nếu số giải pháp nhỏ hơn 3, thì ta tiếp tục chọn vùng ít đông đúc thứ hai và chọn các con sói thủ lĩnh (giải pháp tốt) từ đó.

### 3.3.5 Mã giả MOGWO

---

**Input:** Grey wolf population size  $n$ , Max number of iterations  
**Output:** Non-dominated archive of optimal solutions

---

```

1 : Initialize the grey wolf population  $X_i$  ( $i = 1, 2, \dots, n$ )
2 : Initialize parameters  $a$ ,  $A$ , and  $C$ 
3 : Calculate the objective values for all wolves
4 : Find the non-dominated solutions and initialize the archive with
   them
5 :  $X_\alpha = \text{SelectLeader}(\text{archive})$ .
6 : Exclude alpha from the archive temporarily to avoid selecting the
   same leader
7 :  $X_\beta = \text{SelectLeader}(\text{archive})$ .
8 : Exclude beta from the archive temporarily to avoid selecting the
   same leader
9 :  $X_\delta = \text{SelectLeader}(\text{archive})$ .
10 : Add back  $\alpha$  and  $\beta$  to the archive.
11 :  $t = 1$ .
12 : While ( $t < \text{MaxIter}$ )
13 :   For each search agent
14 :     Update its position using Eqs. 5–7
15 :   End for
16 :   Update  $a$ ,  $A$ , and  $C$ .
17 :   Calculate objective values for all wolves.
18 :   Find the non-dominated solutions.
19 :   Update the archive with respect to the obtained non-dominated
   solutions
20 :   If the archive is full
21 :     Run the grid mechanism to omit one of the current archive
   members
22 :     Add the new solution to the archive.
23 :   End if
24 :   If any of the new added solutions to the archive is located outside
   the hypercubes (region)
25 :     Update grid boundaries to cover the new solutions.
26 :   End if
27 :    $X_\alpha = \text{SelectLeader}(\text{archive})$ .
28 :   Exclude  $\alpha$  from the archive temporarily to avoid selecting the
   same leader
29 :    $X_\beta = \text{SelectLeader}(\text{archive})$ .
30 :   Exclude  $\beta$  from the archive temporarily to avoid selecting the same
   leader
31 :    $X_\delta = \text{SelectLeader}(\text{archive})$ .
32 :   Add back  $\alpha$  and  $\beta$ .
33 :    $t = t + 1$ .
34 : End while
35 : Return archive.

```

---

## 4 References

1. Seyedali Mirjalili, Seyed Mohammad Mirjalili, Andrew Lewis. “Grey Wolf Optimizer.” *Advances in Engineering Software*, Volume 69, March 2014.  
DOI: <https://doi.org/10.1016/j.advengsoft.2013.12.007>
2. Mohammad H. Nadimi-Shahraki, Shokoo Taghian, Seyedali Mirjalili. “An improved Grey Wolf Optimizer for solving engineering problems.” *Expert Systems with Applications*, vol. 166, 2021, Article 113917.  
DOI: <https://doi.org/10.1016/j.eswa.2020.113917>
3. Qiu, Y., Yang, X., Chen, S. “An improved gray wolf optimization algorithm solving functional optimization and engineering design problems.” *Scientific Reports* 14, 14190 (2024).  
DOI: <https://doi.org/10.1038/s41598-024-64526-2>
4. Seyedali Mirjalili, Shahrzad Saremi, Seyed Mohammad Mirjalili, Leandro dos S. Coelho. “Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization.” *Expert Systems with Applications*, Volume 47, 1 April 2016.  
DOI: <https://doi.org/10.1016/j.eswa.2015.10.039>