ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO BÀI TẬP LỚN

TỐI ƯU HÓA BẦY SÓI XÁM
(GREY WOLF OPTIMIZATION)

Nhóm thực hiện: 53

Giảng viên hướng dẫn: TS. Trịnh Văn Chiến

Sinh viên thực hiện: Trần Đăng Bách - 20235661
Phạm Đức Minh - 20235784

Hà Nội, 2026

Mục lục

1 TỔNG QUAN VỀ TỐI ƯU HÓA VÀ TRÍ TUỆ BẦY ĐÀN 4

1.1 Đặt vấn đề tối ưu hóa . . . 4

1.2 Phân loại Meta-heuristics . . . 4

2 TRÌNH BÀY VÀ PHÂN TÍCH THUẬT TOÁN GWO 4

2.1 Cảm hứng . . . 4

2.2 Mô hình phân cấp xã hội . . . 5

2.3 Mô hình toán học của cơ chế săn mồi . . . 5

2.3.1 Bao vây con mồi . . . 5

2.3.2 Săn mồi (Cập nhật vị trí) . . . 6

2.3.3 Cân bằng khám phá và khai thác . . . 6

2.4 Lưu đồ thuật toán (Flowchart) . . . 7

2.5 Mã giả (Detailed Pseudocode) . . . 7

2.6 Phân tích độ phức tạp thuật toán . . . 8

3 CÁC BIẾN THỂ CẢI TIẾN CỦA GWO 9

3.1 Improved GWO (IGWO) . . . 9

3.1.1 Cơ chế Học tập theo chiều (DLH) . . . 9

3.1.2 Cơ chế Lựa chọn Tham lam (Greedy Selection) . . . 10

3.2 Binary GWO (BGWO) . . . 10

3.2.1 Hàm truyền hình chữ S (S-shaped) . . . 10

3.2.2 Hàm truyền hình chữ V (V-shaped) . . . 10

3.3 Chaotic GWO (CGWO) . . . 11

3.3.1 Bản đồ Logistic (Logistic Map) . . . 11

3.3.2 Cơ chế Cập nhật . . . 11

3.4 Hybrid PSO-GWO . . . 11

3.4.1 Cơ chế Cập nhật Vị trí . . . 12

4 MÔ PHỎNG VÀ ĐÁNH GIÁ HIỆU NĂNG 12

4.1 Mô tả bài toán . . . 12

4.2 Thiết lập môi trường thực nghiệm . . . 12

4.3 Phân tích kết quả mô phỏng . . . 13

5 THỰC NGHIỆM TRÊN BÀI TOÁN THỰC TẾ: TỐI ƯU HÓA CNN 14

5.1 Mô tả bài toán và Dữ liệu . . . 14

5.2 Phương pháp thực hiện . . . 14

5.3 Kết quả thực nghiệm . . . 14

5.4 Thảo luận về Kết quả Thực nghiệm . . . 15

6 TỐI ƯU HÓA ĐA CHÙM SÓNG CHO HỆ THỐNG JCAS **16**
6.1 Bối cảnh và Động lực Nghiên cứu 16
6.1.1 Thách thức trong mạng 6G 16
6.1.2 Giải pháp JCAS (Joint Communication and Sensing) 16
6.1.3 Bài toán Trade-off (Cân bằng) 17
6.2 Thiết lập Mục tiêu Tối ưu 17
6.2.1 Hàm mục tiêu (Objective Function) 17
6.2.2 Cấu trúc Pattern mong muốn (P_d) 17
6.2.3 Mô hình Hybrid Beamforming 17
6.3 Mô hình Toán học Hệ thống 18
6.3.1 Biến quyết định (Vector W) 18
6.3.2 Vector đáp ứng mảng (Steering Vector) 18
6.3.3 Thuật toán Two-Step Iterative Least Squares (TS-ILS) 18
6.4 Các Ràng buộc Vật lý và Thiết kế 19
6.4.1 Ràng buộc Công suất (Power Constraint) 19
6.4.2 Ràng buộc Sidelobe (SLL) 19
6.4.3 Ràng buộc Nulls (Triệt tiêu nhiễu) 19
6.4.4 Ràng buộc Đa chùm sóng (Multi-beam Isolation) 20

7 TRIỂN KHAI GWO CHO BÀI TOÁN JCAS BEAMFORMING **20**
7.1 Ánh xạ Bài toán JCAS sang Không gian Tối ưu GWO 20
7.1.1 Mapping các thành phần 20
7.1.2 Không gian tìm kiếm Complex-valued 20
7.2 Cấu hình Thực nghiệm 20
7.2.1 Thiết lập Hyperparameters 20
7.2.2 Thông số Hệ thống JCAS 22

8 KẾT QUẢ THỰC NGHIỆM **22**
8.1 So sánh Fitness theo Kịch bản 23
8.2 Nhận xét Kịch bản Thorough (50 sói, 500 iterations) 23

Tài liệu tham khảo **24**

Danh sách bảng

1 Bảng tổng hợp kết quả thực nghiệm trên CNN 15
2 Ý nghĩa tham số ρ trong ứng dụng thực tế 18
3 Ánh xạ bài toán JCAS sang thuật toán GWO 20
4 Ba kịch bản Hyperparameter Tuning cho GWO variants 21
5 Xếp hạng Fitness cuối cùng theo kịch bản (từ tốt nhất đến kém nhất) 23

Danh sách hình vẽ

1 Lưu đồ thuật toán Grey Wolf Optimization 7
2 Tốc độ hội tụ trên hàm Rastrigin ($D = 30$) 13
3 Biểu đồ so sánh hiệu năng giữa Random CNN và GWO-CNN 15
4 So sánh hiệu năng các thuật toán trên ba kịch bản: Fast (20 sói, 100 iterations), Balanced (30 sói, 300 iterations), và Thorough (50 sói, 500 iterations). Hàng trên: đường cong hội tụ fitness. Hàng dưới: beam pattern của nghiệm tốt nhất. 22

1 TỔNG QUAN VỀ TỐI ƯU HÓA VÀ TRÍ TUỆ BẦY ĐÀN

1.1 Đặt vấn đề tối ưu hóa

Trong khoa học máy tính và kỹ thuật, tối ưu hóa là quá trình tìm kiếm một giải pháp tốt nhất từ tập hợp các giải pháp khả thi. Một bài toán tối ưu hóa tổng quát có thể được biểu diễn dưới dạng toán học như sau:

$$\text{Minimize/Maximize } F(\vec{x}) \tag{1}$$

Thỏa mãn các ràng buộc:

$$g_j(\vec{x}) \leq 0, \quad j = 1, \ldots, m \tag{2}$$

$$h_k(\vec{x}) = 0, \quad k = 1, \ldots, p \tag{3}$$

Trong đó $\vec{x} = (x_1, x_2, \ldots, x_n)$ là véc-tơ các biến quyết định.

1.2 Phân loại Meta-heuristics

Meta-heuristics có thể được chia thành hai nhánh chính:

1. **Dựa trên tiến hóa (Evolutionary Algorithms - EA):** Đại diện tiêu biểu là Giải thuật di truyền (Genetic Algorithm - GA).
2. **Trí tuệ bầy đàn (Swarm Intelligence - SI):** Lấy cảm hứng từ hành vi tập thể của các sinh vật trong tự nhiên như đàn chim, đàn cá, đàn kiến. GWO thuộc nhóm này.

2 TRÌNH BÀY VÀ PHÂN TÍCH THUẬT TOÁN GWO

2.1 Cảm hứng

Nguồn cảm hứng của thuật toán GWO bắt nguồn từ tập tính xã hội và kỹ thuật săn mồi của loài **Sói xám** (*Canis lupus*). Đây là loài động vật săn mồi đỉnh cao (apex predators), đứng đầu chuỗi thức ăn và thường sống theo bầy đàn với quy mô trung bình từ 5 đến 12 cá thể.

Hai đặc điểm thú vị nhất của loài sói xám tạo nên nền tảng cho thuật toán này là:

- **Trí tuệ bầy đàn (Swarm Intelligence):** Không giống như các loài săn mồi đơn độc, sói xám phối hợp cực kỳ ăn ý. Sự thành công của chúng không đến từ sức mạnh cá nhân mà đến từ kỷ luật và sự phân công lao động.
- **Quy trình săn mồi đa giai đoạn:** Quá trình săn mồi của sói xám trong tự nhiên luôn tuân theo một kịch bản chặt chẽ gồm 3 bước chính:
    1. *Truy vết và Tiếp cận:* Bầy sói tìm kiếm và bám theo con mồi (tương ứng với giai đoạn Khám phá - Exploration trong toán học).

2. *Bao vây và Quấy rối:* Khi đã xác định được mục tiêu, chúng bao vây từ nhiều phía để cô lập con mồi, khiến con mồi kiệt sức.
3. *Tấn công:* Khi con mồi đã yếu, các con đầu đàn sẽ ra lệnh tấn công để kết liễu (tương ứng với giai đoạn Khai thác - Exploitation).

2.2 Mô hình phân cấp xã hội

Trong thuật toán GWO, mô hình xã hội của loài sói được toán học hóa để xây dựng cơ chế lựa chọn giải pháp. Bầy sói tuân theo một hệ thống phân cấp quyền lực hình kim tự tháp nghiêm ngặt, bao gồm 4 tầng lớp:

1. **Alpha (α) - Lãnh đạo tối cao:**
   - Trong tự nhiên: Là con sói đực hoặc cái dẫn đầu, chịu trách nhiệm ra quyết định về nơi ngủ, thời điểm săn mồi và hướng di chuyển. Alpha không nhất thiết phải là con mạnh nhất, mà là con có khả năng quản lý tốt nhất.
   - Trong GWO: Đây là **giải pháp tốt nhất** (Best Solution) tìm được trong quần thể hiện tại. Mọi mệnh lệnh cập nhật vị trí đều xoay quanh vị trí của Alpha.
2. **Beta (β) - Cố vấn chiến lược:**
   - Trong tự nhiên: Là cấp dưới trực tiếp của Alpha. Beta đóng vai trò cố vấn, thi hành kỷ luật với cấp dưới và là ứng cử viên thay thế khi Alpha già yếu hoặc qua đời.
   - Trong GWO: Đây là **giải pháp tốt thứ hai**. Beta hỗ trợ Alpha trong việc định hướng không gian tìm kiếm.
3. **Delta (δ) - Lực lượng chuyên trách:**
   - Trong tự nhiên: Nhóm này bao gồm các ”chuyên gia” như: *Trinh sát* (Scouts) chuyên canh gác, *Lính canh* (Sentinels) bảo vệ bầy, và *Thợ săn* (Hunters). Chúng thống trị Omega nhưng tuân lệnh Alpha và Beta.
   - Trong GWO: Đây là **giải pháp tốt thứ ba**. Cùng với α và β, δ tạo thành bộ ba dẫn đường.
4. **Omega (ω) - Lực lượng thực thi:**
   - Trong tự nhiên: Là tầng lớp thấp nhất, đóng vai trò ”vùng đệm” giảm căng thẳng và duy trì cấu trúc bầy.
   - Trong GWO: Đây là **các giải pháp ứng viên còn lại**. Chúng không có quyền ra quyết định mà buộc phải cập nhật vị trí dựa trên trung bình cộng vị trí của α, β và δ.

2.3 Mô hình toán học của cơ chế săn mồi

2.3.1 Bao vây con mồi

Khi sói xám xác định được vị trí con mồi, chúng bắt đầu bao vây. Hành vi này được mô hình hóa bằng các phương trình:

$$D = |C \cdot X_p(t) - X(t)| \tag{4}$$

$$X(t+1) = X_p(t) - A \cdot D \tag{5}$$

Trong đó:

- t: Vòng lặp hiện tại.
- X_p: Véc-tơ vị trí của con mồi (có thể là vị trí X_α).
- X: Véc-tơ vị trí của một con sói.
- D: Véc-tơ khoảng cách đã điều chỉnh.
- A và C: Các véc-tơ hệ số.

2.3.2 Săn mồi (Cập nhật vị trí)

GWO giả định rằng Alpha, Beta và Delta có thông tin tốt nhất về vị trí con mồi. Các sói Omega cập nhật vị trí dựa trên vị trí trung bình của ba nhà lãnh đạo này.

Đầu tiên, tính toán các bước di chuyển giả định đối với từng con đầu đàn:

$$D_\alpha = |C_1 \cdot X_\alpha - X| \quad \Rightarrow \quad X_1 = X_\alpha - A_1 \cdot D_\alpha \tag{6}$$

$$D_\beta = |C_2 \cdot X_\beta - X| \quad \Rightarrow \quad X_2 = X_\beta - A_2 \cdot D_\beta \tag{7}$$

$$D_\delta = |C_3 \cdot X_\delta - X| \quad \Rightarrow \quad X_3 = X_\delta - A_3 \cdot D_\delta \tag{8}$$

Sau đó, vị trí mới của con sói Omega là trung bình cộng của ba vị trí trên:

$$X(t+1) = \frac{X_1 + X_2 + X_3}{3} \tag{9}$$

Về mặt hình học, vị trí mới là một điểm ngẫu nhiên nằm bên trong "tam giác" được tạo bởi $X_\alpha, X_\beta, X_\delta$. Cơ chế này cho phép các con sói Omega di chuyển dần về phía vùng không gian hứa hẹn nhất.

2.3.3 Cân bằng khám phá và khai thác

Để thuật toán hoạt động hiệu quả, cần có sự cân bằng giữa tìm kiếm toàn cục (khám phá - exploration) và tinh chỉnh cục bộ (khai thác - exploitation). Các hệ số A và C đóng vai trò quyết định:

$$A = 2a \cdot r_1 - a \tag{10}$$

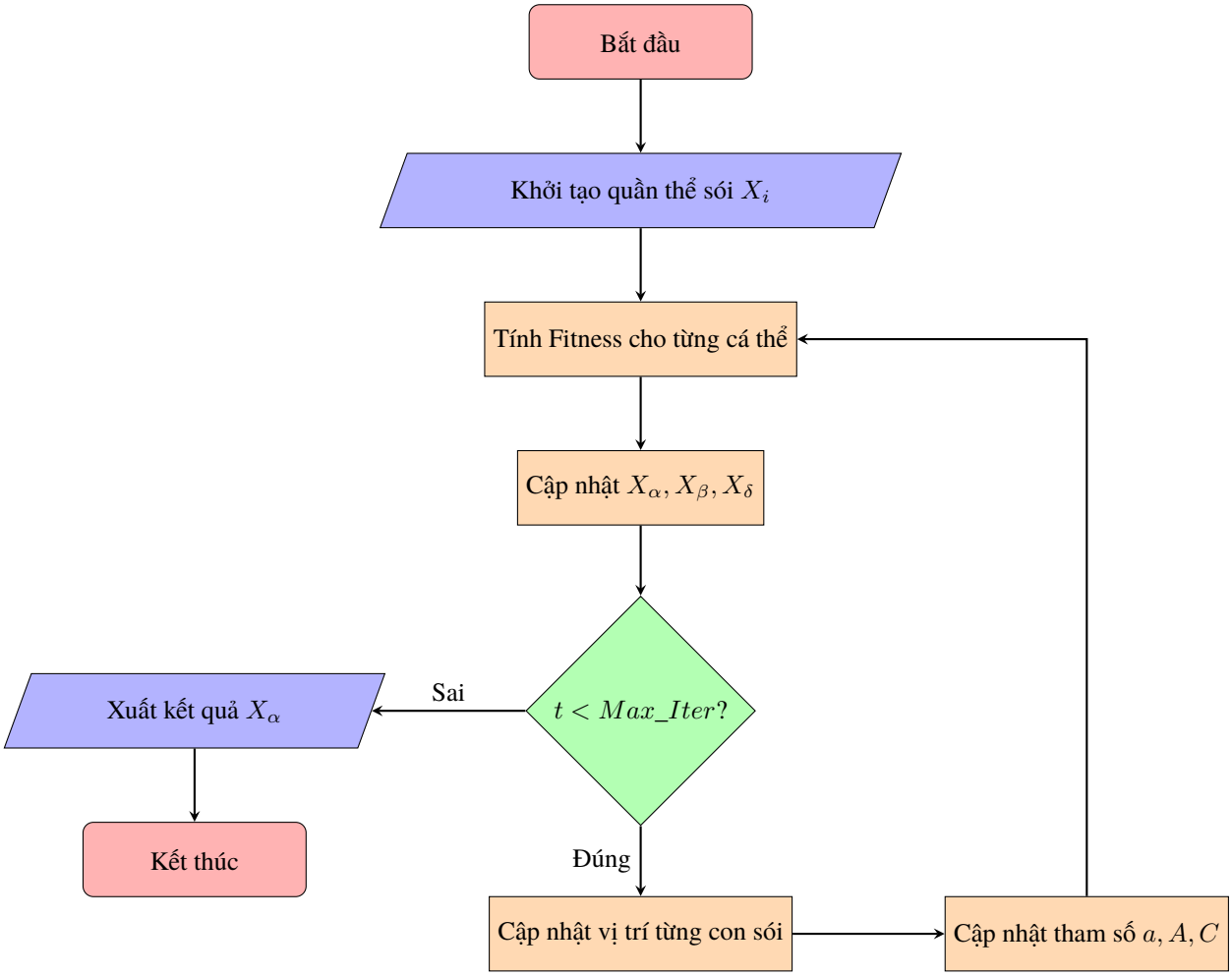
$$C = 2 \cdot r_2 \tag{11}$$

Trong đó r_1, r_2 là các véc-tơ ngẫu nhiên trong đoạn $[0, 1]$. Tham số a giảm tuyến tính từ 2 về 0 trong suốt quá trình lặp.

- **Khám phá (Exploration):** Xảy ra khi $|A| > 1$. Lúc này, con sói bị buộc phải di chuyển ra xa vị trí con mồi (hoặc các sói đầu đàn), giúp mở rộng phạm vi tìm kiếm và tránh tối ưu cục bộ.
- **Khai thác (Exploitation):** Xảy ra khi $|A| < 1$. Lúc này, con sói hội tụ về phía con mồi để tấn công.

Ngoài ra, véc-tơ C chứa giá trị ngẫu nhiên trong $[0, 2]$. C đóng vai trò là trọng số ngẫu nhiên cho vị trí con mồi, mô phỏng các chướng ngại vật tự nhiên khiến sói khó tiếp cận, giúp thuật toán duy trì tính ngẫu nhiên ngay cả ở các vòng lặp cuối.

2.4 Lưu đồ thuật toán (Flowchart)



Hình 1: Lưu đồ thuật toán Grey Wolf Optimization

2.5 Mã giả (Detailed Pseudocode)

Dưới đây là mã giả chi tiết của GWO, mô tả cụ thể từng bước tính toán véc-tơ để cập nhật vị trí.

Algorithm 1 Grey Wolf Optimization (GWO)

Đầu vào: Quần thể size N, Số vòng lặp $MaxIt$, Hàm mục tiêu $f(x)$
Đầu ra: Vị trí tối ưu X_α, Giá trị tối ưu $Score_\alpha$

Bước 1: Khởi tạo
$X_i \leftarrow$ random initialization $(i = 1, \ldots, N)$ // *Khởi tạo vị trí ngẫu nhiên*
$a, A, C \leftarrow$ initial values
$X_\alpha, X_\beta, X_\delta \leftarrow \vec{0}$ // *Khởi tạo vị trí lãnh đạo*
$Score_\alpha, Score_\beta, Score_\delta \leftarrow \infty$ // *Khởi tạo điểm số (Minimization)*
$t \leftarrow 0$
Bước 2: Vòng lặp chính
while $t < MaxIt$ **do**
    // Cập nhật độ thích nghi và xếp hạng sói
    for each wolf $i \in 1 \ldots N$ **do**
        Calculate $fitness = f(X_i)$
        if $fitness < Score_\alpha$ **then**
            $Score_\alpha \leftarrow fitness$, $X_\alpha \leftarrow X_i$
        else if $fitness < Score_\beta$ **then**
            $Score_\beta \leftarrow fitness$, $X_\beta \leftarrow X_i$
        else if $fitness < Score_\delta$ **then**
            $Score_\delta \leftarrow fitness$, $X_\delta \leftarrow X_i$
        end if
    end for
    // Cập nhật vị trí các sói Omega
    Update a linearly from 2 to 0
    for each wolf $i \in 1 \ldots N$ **do**
        for each dimension $d \in 1 \ldots Dim$ **do**
            $r_1, r_2 \leftarrow rand[0, 1]$
            // Tính toán dựa trên Alpha
            $A_1 = 2a \cdot r_1 - a$, $C_1 = 2 \cdot r_2$
            $D_\alpha = |C_1 \cdot X_{\alpha,d} - X_{i,d}|$
            $X_1 = X_{\alpha,d} - A_1 \cdot D_\alpha$
            // Tính toán dựa trên Beta
            $A_2 = 2a \cdot r_1 - a$, $C_2 = 2 \cdot r_2$
            $D_\beta = |C_2 \cdot X_{\beta,d} - X_{i,d}|$
            $X_2 = X_{\beta,d} - A_2 \cdot D_\beta$
            // Tính toán dựa trên Delta
            $A_3 = 2a \cdot r_1 - a$, $C_3 = 2 \cdot r_2$
            $D_\delta = |C_3 \cdot X_{\delta,d} - X_{i,d}|$
            $X_3 = X_{\delta,d} - A_3 \cdot D_\delta$
            // Cập nhật vị trí mới
            $X_{i,d}(t+1) = (X_1 + X_2 + X_3)/3$
        end for
    end for
    // Xử lý biên
    Clip X_i within $[LB, UB]$
    $t \leftarrow t + 1$
end while
return $X_\alpha, Score_\alpha$

2.6 Phân tích độ phức tạp thuật toán

Độ phức tạp tính toán là một tiêu chí quan trọng để đánh giá hiệu năng của thuật toán. Đối với GWO, độ phức tạp phụ thuộc vào số lượng sói (N), số chiều của bài toán (D) và số vòng lặp tối đa (T_{max}).

Quá trình tính toán bao gồm các giai đoạn sau:

1. **Khởi tạo quần thể:** Cần khởi tạo ngẫu nhiên vị trí cho N con sói trong không gian D chiều. Độ phức tạp là

$O(N \times D)$.

2. **Đánh giá hàm mục tiêu (Fitness evaluation):** Trong mỗi vòng lặp, ta cần tính độ thích nghi cho từng con sói. Giả sử chi phí tính hàm mục tiêu là C, độ phức tạp là $O(T_{max} \times N \times C)$.

3. **Cập nhật vị trí:** Với mỗi con sói, thuật toán thực hiện cập nhật vị trí trên từng chiều dựa trên 3 con đầu đàn. Độ phức tạp là $O(T_{max} \times N \times D)$.

Tổng hợp lại, độ phức tạp thời gian (Time Complexity) của GWO được ước tính là:

$$O(GWO) = O(N \cdot D) + O(T_{max} \cdot N \cdot (C + D)) \tag{12}$$

Do C và D thường nhỏ hơn nhiều so với T_{max}, ta có thể xấp xỉ độ phức tạp là $O(T_{max} \cdot N \cdot D)$.

Về độ phức tạp không gian (Space Complexity), GWO cần lưu trữ vị trí hiện tại của N con sói, mỗi con có D chiều. Do đó:

$$Space(GWO) = O(N \cdot D) \tag{13}$$

Điều này cho thấy GWO là một thuật toán hiệu quả về mặt bộ nhớ, phù hợp để triển khai trên các hệ thống nhúng có tài nguyên hạn chế.

3 CÁC BIẾN THỂ CẢI TIẾN CỦA GWO

3.1 Improved GWO (IGWO)

Trong phiên bản GWO gốc, cấu trúc phân cấp nghiêm ngặt khiến toàn bộ bầy sói Omega chỉ di chuyển theo hướng của ba con đầu đàn (α, β, δ). Điều này tuy giúp thuật toán hội tụ nhanh nhưng lại làm giảm đáng kể sự đa dạng của quần thể (diversity), dẫn đến nguy cơ cao bị mắc kẹt tại các cực trị địa phương (local optima).

Biến thể Improved GWO (IGWO), điển hình là phiên bản đề xuất bởi Nadimi-Shahraki et al. (2020), khắc phục nhược điểm này bằng cách giới thiệu một cơ chế lai ghép thông minh gọi là Dimension Learning-based Hunting (DLH).

3.1.1 Cơ chế Học tập theo chiều (DLH)

DLH cho phép các con sói không chỉ học hỏi từ lãnh đạo mà còn học hỏi từ những "người hàng xóm" của chúng. Quá trình này mô phỏng việc chia sẻ thông tin ngang hàng trong bầy đàn. Công thức cập nhật vị trí mới dựa trên DLH được thực hiện trên từng chiều d của vector vị trí:

$$X_{i-DLH,d}(t+1) = X_{i,d}(t) + rand \cdot (X_{n,d}(t) - X_{r,d}(t)) \tag{14}$$

Trong đó:

- $X_{i,d}(t)$: Vị trí hiện tại của con sói thứ i ở chiều d.
- $X_{n,d}(t)$: Vị trí của một con sói lân cận (Neighbor). Con sói lân cận này được chọn từ quần thể dựa trên khoảng cách Euclide gần nhất với con sói i, đảm bảo tính cục bộ của thông tin.

- $X_{r,d}(t)$: Vị trí của một con sói được chọn ngẫu nhiên từ quần thể.

Việc kết hợp thông tin từ hàng xóm và cá thể ngẫu nhiên giúp duy trì sự đa dạng gen, ngăn chặn sự hội tụ sớm.

3.1.2 Cơ chế Lựa chọn Tham lam (Greedy Selection)

Sau khi tính toán được vị trí tiềm năng từ DLH (X_{DLH}), thuật toán sẽ không chấp nhận ngay lập tức. Nó so sánh độ thích nghi (fitness) của vị trí mới này với vị trí sinh ra bởi cơ chế GWO truyền thống (X_{GWO}).

$$X_i(t+1) = \begin{cases} X_{GWO} & \text{nếu } f(X_{GWO}) < f(X_{DLH}) \\ X_{DLH} & \text{ngược lại} \end{cases} \tag{15}$$

Cơ chế này đảm bảo rằng con sói luôn di chuyển đến vị trí tốt hơn, tận dụng được ưu điểm của cả hai chiến lược: khả năng hội tụ nhanh của GWO và khả năng khám phá mạnh mẽ của DLH.

3.2 Binary GWO (BGWO)

Thuật toán GWO nguyên bản được thiết kế cho không gian tìm kiếm liên tục (số thực). Tuy nhiên, nhiều bài toán thực tế như Lựa chọn đặc trưng (Feature Selection) hay Bài toán cái túi (Knapsack Problem) yêu cầu không gian tìm kiếm rời rạc (nhị phân 0 hoặc 1). BGWO ra đời để giải quyết vấn đề này.

Trong BGWO, vị trí của mỗi con sói là một vector bit. Việc cập nhật vị trí không thể thực hiện bằng các phép cộng trừ vector thông thường. Thay vào đó, BGWO sử dụng khái niệm Hàm truyền (Transfer Function) để chuyển đổi giá trị thực thành xác suất thay đổi bit.

3.2.1 Hàm truyền hình chữ S (S-shaped)

Đây là cách tiếp cận phổ biến nhất, sử dụng hàm Sigmoid để ép giá trị vận tốc về khoảng $[0, 1]$.

$$T(x^d) = \frac{1}{1 + e^{-10(x^d - 0.5)}} \tag{16}$$

Quy tắc cập nhật vị trí bit tại chiều d ở vòng lặp $t + 1$:

$$X_d(t+1) = \begin{cases} 1 & \text{nếu } rand < T(x^d) \\ 0 & \text{ngược lại} \end{cases} \tag{17}$$

Trong đó $rand$ là một số ngẫu nhiên trong khoảng $[0, 1]$. Nếu xác suất chuyển đổi $T(x^d)$ cao, khả năng bit đó trở thành 1 là lớn.

3.2.2 Hàm truyền hình chữ V (V-shaped)

Một cách tiếp cận khác hiệu quả hơn là sử dụng hàm truyền hình chữ V (dựa trên hàm $tanh$). Hàm V-shaped không ép buộc bit về 0 hay 1 mà quyết định xem bit đó có nên giữ nguyên hay đảo trạng thái (complement) hay không.

$$T_V(x^d) = |\tanh(x^d)| \tag{18}$$

Nếu giá trị bước nhảy nhỏ (gần 0), hàm T_V trả về giá trị thấp, nghĩa là con sói nên giữ nguyên vị trí bit hiện tại để khai thác cục bộ, thay vì nhảy lung tung. Điều này giúp BGWO ổn định hơn so với S-shaped.

3.3 Chaotic GWO (CGWO)

Trong thuật toán GWO tiêu chuẩn, sự cân bằng giữa khám phá và khai thác phụ thuộc lớn vào tham số a (giảm tuyến tính từ 2 về 0) và các véc-tơ ngẫu nhiên r_1, r_2 (phân phối đều). Tuy nhiên, tính ngẫu nhiên đơn thuần này đôi khi không đủ "động lực" để giúp bầy sói thoát khỏi các cực trị địa phương bền vững, đặc biệt trong các bài toán đa cực trị phức tạp.

Biến thể **Chaotic GWO (CGWO)** tích hợp lý thuyết hỗn loạn (Chaos Theory) vào quá trình cập nhật vị trí. Sự hỗn loạn là trạng thái phi tuyến tính, nhạy cảm với điều kiện đầu, giúp thuật toán duy trì sự đa dạng tốt hơn so với số ngẫu nhiên truyền thống.

3.3.1 Bản đồ Logistic (Logistic Map)

Trong CGWO, các số ngẫu nhiên r_1, r_2 thường được thay thế bằng các chuỗi số sinh ra từ bản đồ hỗn loạn. Bản đồ Logistic là loại phổ biến nhất do tính đơn giản và hiệu quả cao:

$$Ch_{t+1} = \mu \cdot Ch_t \cdot (1 - Ch_t) \tag{19}$$

Trong đó:

- Ch_t: Giá trị hỗn loạn ở vòng lặp thứ t, nằm trong khoảng $(0, 1)$.
- μ: Tham số điều khiển. Khi $\mu = 4$, hệ thống đạt trạng thái hỗn loạn hoàn toàn.

3.3.2 Cơ chế Cập nhật

Thay vì sử dụng r_1 ngẫu nhiên, véc-tơ hệ số A được tính lại dựa trên giá trị hỗn loạn:

$$A = 2a \cdot Ch_t - a \tag{20}$$

Nhờ tính chất ergodic (đi qua mọi trạng thái) của hàm hỗn loạn, giá trị A sẽ biến thiên phong phú hơn, giúp bầy sói thực hiện các bước nhảy vọt cần thiết để thoát khỏi bẫy cục bộ.

3.4 Hybrid PSO-GWO

Thuật toán Tối ưu hóa Bầy sói xám (GWO) tiêu chuẩn gặp phải những thách thức đáng kể khi áp dụng vào các không gian phức tạp và nhiều chiều. Hai hạn chế chính là: (1) xu hướng bị mắc kẹt tại các cực trị địa phương trong giai đoạn sau của quá trình tối ưu hóa do khả năng khám phá kém, và (2) thiếu sự đa dạng quần thể do phương pháp khởi tạo đồng nhất tiêu chuẩn [1]. Zhang và cộng sự (2021) cùng Kumar và cộng sự (2022) đã giới thiệu các cơ chế hybrid kết hợp khả năng khai thác của GWO với sức mạnh khám phá của PSO.

3.4.1 Cơ chế Cập nhật Vị trí

Trong GWO tiêu chuẩn, vị trí của một con sói được cập nhật dựa trên vị trí trung bình của ba con đầu đàn (α, β, δ). Tuy nhiên, cách tiếp cận này có thể dẫn đến hội tụ sớm. Cơ chế lai do Kumar và cộng sự (2022) đề xuất đã đưa thêm thành phần vận tốc V_i^{t+1} vào phương trình cập nhật vị trí, được định nghĩa là:

$$V_i^{t+1} = w(t) \cdot V_i^t + c_1 \cdot r_1 \cdot (\mathbf{p}_{best,i} - \mathbf{X}_i^t) + c_2 \cdot r_2 \cdot (\mathbf{g}_{best} - \mathbf{X}_i^t) \tag{21}$$

trong đó $w(t)$ là trọng số quán tính thích nghi giảm tuyến tính từ w_{max} xuống w_{min} để cân bằng giữa tìm kiếm toàn cục và cục bộ. Việc cập nhật vị trí cuối cùng là một tổ hợp có trọng số của các thành phần GWO và PSO:

$$\mathbf{X}_i^{t+1} = \omega_{GWO} \cdot \left(\frac{\mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3}{3} \right) + \omega_{PSO} \cdot (\mathbf{X}_i^t + \mathbf{V}_i^{t+1}) \tag{22}$$

Không giống như gánh nặng triển khai lớn của MOGWO hay việc xây dựng vùng lân cận phức tạp của I-GWO, PSO-GWO Hybrid mang lại sự đơn giản đáng kể.

Một lợi thế quan trọng của PSO-GWO Hybrid là nó đưa ra một giải pháp duy nhất, làm cho nó có thể so sánh trực tiếp với Bình phương Tối thiểu Lặp lại Hai bước (TS-ILS) và các phương pháp cơ sở JCAS chuẩn khác. Không cần hậu xử lý bổ sung—không cần lựa chọn điểm gập, không cần khớp trọng số, không cần tính toán siêu thể tích. Tất cả các chỉ số tiêu chuẩn (giá trị mục tiêu cuối cùng, SINR truyền thông, MI cảm biến, PSLL, số vòng lặp hội tụ, thời gian tính toán) đều có thể được so sánh trực tiếp.

Mặc dù kết hợp cả hai cơ chế GWO và PSO, chi phí tính toán phụ trội là rất nhỏ. Phân tích độ phức tạp thời gian cho thấy: cập nhật vị trí là $O(N \times D)$ (giống như GWO), cập nhật vận tốc là $O(N \times D)$ (chi phí PSO), và tổng mỗi vòng lặp là $O(2N \times D)$ [2, 3].

4 MÔ PHỎNG VÀ ĐÁNH GIÁ HIỆU NĂNG

4.1 Mô tả bài toán

Nhóm thực hiện mô phỏng trên hàm Rastrigin. Đây là hàm mục tiêu có độ phức tạp cao với hàng ngàn điểm cực trị địa phương. Công thức toán học của hàm Rastrigin cho n chiều:

$$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)] \tag{23}$$

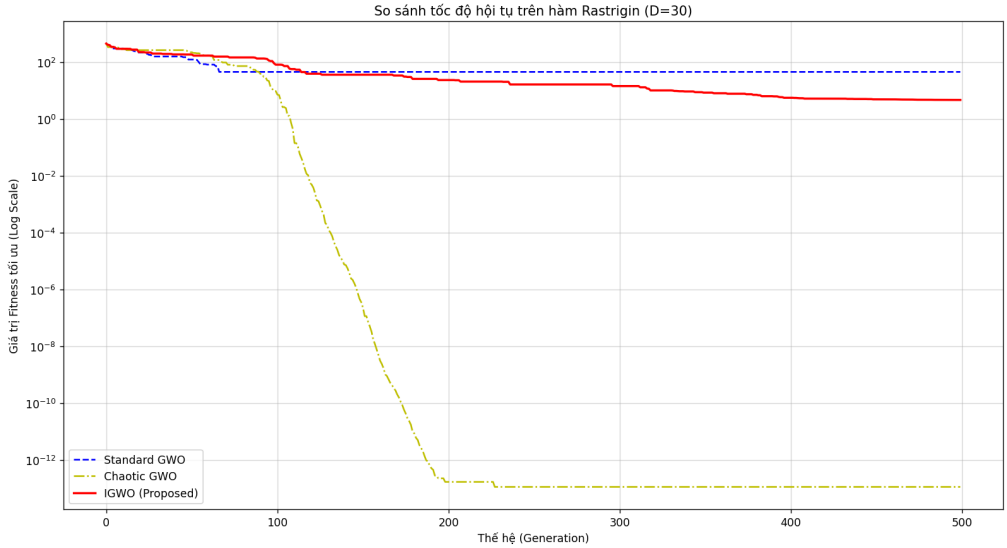
Trong đó $x_i \in [-5.12, 5.12]$. Giá trị tối ưu toàn cục là 0 tại $x = \vec{0}$.

4.2 Thiết lập môi trường thực nghiệm

- **Số chiều:** $D = 30$.
- **Kích thước quần thể:** $N = 50$.
- **Số vòng lặp:** $T_{max} = 500$.

- **So sánh:** GWO gốc, Chaotic GWO (CGWO), Improved GWO (IGWO).

4.3 Phân tích kết quả mô phỏng



Hình 2: Tốc độ hội tụ trên hàm Rastrigin ($D = 30$)

Dựa vào đồ thị mô phỏng thực tế tại Hình 2 (với trục tung sử dụng thang đo Logarit), ta có những quan sát trái ngược với dự đoán lý thuyết ban đầu và đưa ra các phân tích kỹ thuật như sau:

1. Đối với GWO Gốc (Standard GWO) - Đường màu xanh nét đứt:

- **Hiện tượng:** Thuật toán giảm lỗi nhanh trong 50 thế hệ đầu nhưng lập tức đi ngang (flatline) ở mức 10^2.
- **Đánh giá:** Kết quả này khẳng định nhược điểm cố hữu của GWO là *ngưng trệ sớm (premature convergence)*. Trên hàm đa cực trị như Rastrigin, ba con đầu đàn bị kẹt ở các cực trị địa phương kém chất lượng, kéo theo toàn bộ bầy Omega không thể thoát ra.

2. Đối với Improved GWO (IGWO) - Đường màu đỏ nét liền:

- **Hiện tượng:** IGWO cho thấy sự giảm dần đều đặn nhưng tốc độ rất chậm và kết thúc ở mức giá trị fitness khá cao (lớn hơn 1). Nó không đạt được khả năng hội tụ sâu về 0.
- **Đánh giá:** Mặc dù cơ chế DLH (Dimension Learning-based Hunting) giúp tăng tính đa dạng, nhưng cơ chế *Lựa chọn Tham lam (Greedy Selection)* đã kìm hãm thuật toán. Trên địa hình "gồ ghề" của Rastrigin, việc từ chối các bước di chuyển tồi hơn (nhưng cần thiết để thoát hố) khiến bầy sói bị kẹt. Ngoài ra, việc học hỏi từ hàng xóm (Neighbors) trong không gian 30 chiều có thể đã gây nhiễu khi các hàng xóm cũng đang nằm ở vùng tối ưu cục bộ.

3. Đối với Chaotic GWO (CGWO) - Đường màu vàng nét gạch chấm:

- **Hiện tượng:** Đây là thuật toán hiệu quả nhất trong thực nghiệm này. Sau giai đoạn đầu tìm kiếm chậm, thuật toán có sự giảm đột ngột (đổ đèo) trong khoảng thế hệ 100-200 và đạt độ chính xác cực cao (10^{-13}).

- **Đánh giá:** Sự thành công này đến từ việc ứng dụng *Bản đồ hỗn loạn (Chaotic Map)*. Tính chất phi tuyến tính và ngẫu nhiên mạnh của chuỗi số hỗn loạn giúp tạo ra các bước nhảy vọt (jumps), cho phép bầy sói thoát khỏi các "bẫy" cực trị địa phương mà GWO thường và IGWO đều bị mắc kẹt.

5 THỰC NGHIỆM TRÊN BÀI TOÁN THỰC TẾ: TỐI ƯU HÓA CNN

Bên cạnh các hàm toán học chuẩn, nhóm thực hiện đã áp dụng GWO vào một bài toán thực tế trong lĩnh vực Học sâu (Deep Learning): Tối ưu hóa siêu tham số cho mạng CNN nhận diện chữ số viết tay.

5.1 Mô tả bài toán và Dữ liệu

- Mục tiêu: Tìm bộ siêu tham số (Hyperparameters) tối ưu bao gồm *Learning Rate* và *Số lượng bộ lọc (Filters)* để tối đa hóa độ chính xác của mạng CNN.
- Dữ liệu: Tập ảnh chữ số viết tay (0-9). Tổng cộng có 21,555 ảnh đen trắng, kích thước gốc được resize về 28×28 pixels.
- Công cụ thực hiện: Python, TensorFlow

5.2 Phương pháp thực hiện

Hai phương pháp được triển khai để so sánh hiệu năng:

Phương pháp 1: CNN với Hyperparameters ngẫu nhiên (Random Search)

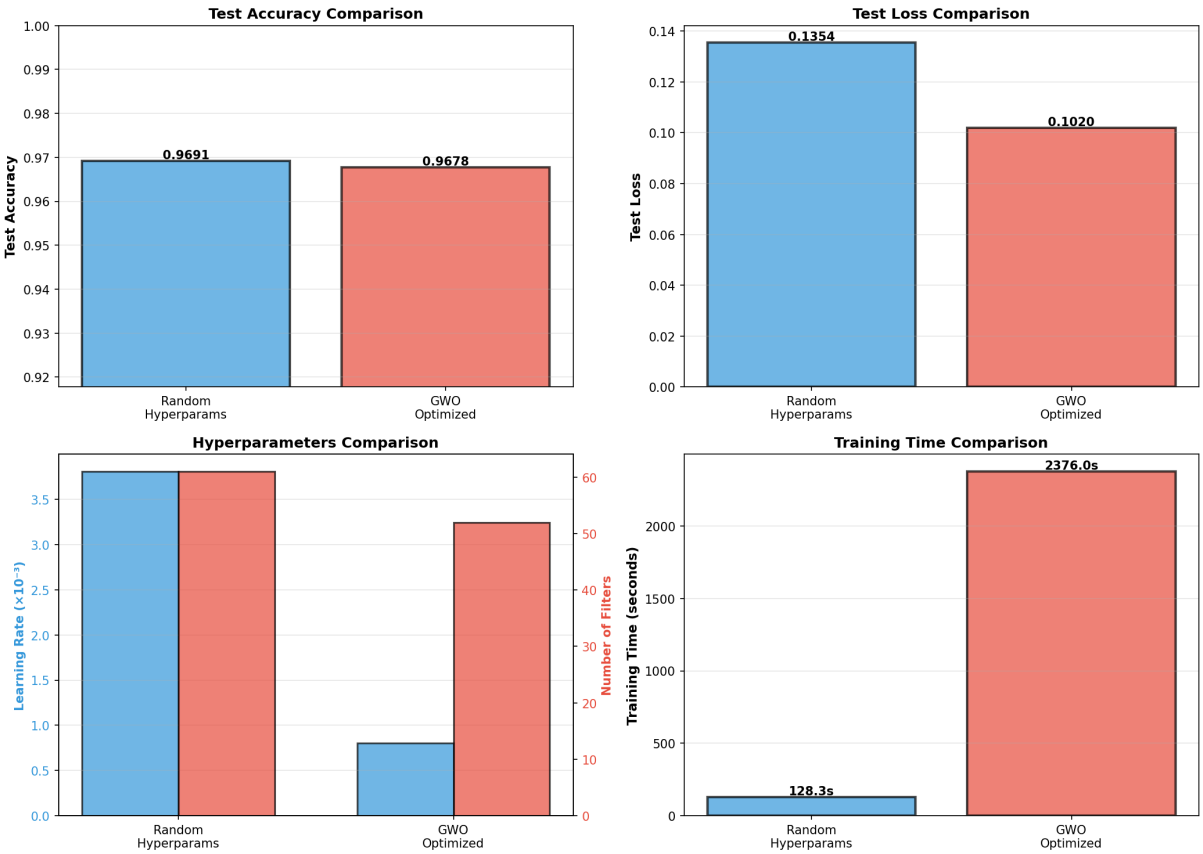
- Khởi tạo Learning Rate ngẫu nhiên trong khoảng $[0.0001, 0.01]$.
- Khởi tạo Số lượng Filters ngẫu nhiên trong khoảng $[16, 64]$.
- Huấn luyện mô hình trong 30 epochs.

Phương pháp 2: CNN với GWO Optimization

- Không gian tìm kiếm: Tương tự như trên.
- Hàm mục tiêu (Fitness Function): Giá trị *Validation Loss* sau khi train nhanh (15 epochs) với cơ chế Early Stopping.
- Cấu hình GWO: Số lượng sói $N = 5$, Số vòng lặp tối ưu hóa $T = 8$.
- Sau khi tìm được bộ tham số tối ưu, mô hình được huấn luyện lại đầy đủ trong 30 epochs.

5.3 Kết quả thực nghiệm

Sau quá trình chạy thực nghiệm, nhóm thu được kết quả chi tiết như sau:



Hình 3: Biểu đồ so sánh hiệu năng giữa Random CNN và GWO-CNN

Bảng 1: Bảng tổng hợp kết quả thực nghiệm trên CNN

Chỉ số	**Random CNN**	**GWO Optimized CNN**
Learning Rate	0.003808	0.000802
Num Filters	61	52
Test Accuracy	96.91%	96.78%
Test Loss	0.1354	**0.1020**

Phân tích và Đánh giá:

- Về độ chính xác (Accuracy): Hai phương pháp cho kết quả tương đương nhau ($\sim 96.8\%$). Random CNN may mắn tìm được cấu hình tốt ngay từ đầu nên nhỉnh hơn một chút (0.14%) không đáng kể.
- Về độ mất mát (Loss): GWO đã tìm ra bộ tham số giúp mô hình giảm Validation Loss xuống còn 0.1020 (so với 0.1354 của Random), tương ứng mức cải thiện 24.70%. Điều này cho thấy mô hình tối ưu bởi GWO có độ tin cậy cao hơn trong các dự đoán và ít bị Overfitting hơn.

5.4 Thảo luận về Kết quả Thực nghiệm

Phương pháp GWO tạo ra mô hình có độ mất mát (Test Loss) thấp hơn đáng kể (0.1020 so với 0.1354) mặc dù độ chính xác (Accuracy) thấp hơn không đáng kể (96.78% so với 96.91%) so với phương pháp ngẫu nhiên.

Điều này dẫn đến các nhận định sau:

1. **Tính Bền vững của Mô hình (Model Robustness):** Việc GWO đạt được Loss thấp hơn cho thấy mô hình này có độ tin cậy cao hơn trong các dự đoán. Các vector xác suất đầu ra (Softmax probabilities) của mô hình GWO hội tụ gần về 1 (cho nhãn đúng) và 0 (cho nhãn sai) hơn so với mô hình Random. Điều này đồng nghĩa với việc mô hình GWO ít bị "lưỡng lự" tại các biên quyết định, hứa hẹn khả năng chống nhiễu và tổng quát hóa tốt hơn trong môi trường thực tế.

2. **Chiến lược Tối ưu hóa:** GWO đã hội tụ về một vùng tham số với Learning Rate nhỏ (8×10^{-4}), cho thấy thuật toán ưu tiên khả năng tinh chỉnh trọng số chi tiết (Fine-tuning) để cực tiểu hóa hàm mục tiêu (Loss). Trong khi đó, Random Search hưởng lợi từ việc khởi tạo ngẫu nhiên một Learning Rate lớn (3.8×10^{-3}), giúp thoát khỏi các vùng bằng phẳng nhanh chóng nhưng có nguy cơ dao động quanh điểm tối ưu toàn cục.

6 TỐI ƯU HÓA ĐA CHÙM SÓNG CHO HỆ THỐNG JCAS

6.1 Bối cảnh và Động lực Nghiên cứu

6.1.1 Thách thức trong mạng 6G

Hệ thống thông tin vô tuyến thế hệ mới (6G) đối mặt với mâu thuẫn giữa nhu cầu truyền dẫn dữ liệu tốc độ cao và yêu cầu cảm biến môi trường chính xác.

Mô hình truyền thống tách biệt hai hệ thống riêng lẻ (Communication và Sensing) bộc lộ các nhược điểm nghiêm trọng:

- **Chi phí (Cost):** Tăng gấp đôi gánh nặng đầu tư phần cứng, vận hành và bảo trì.
- **Kích thước (Size):** Thiết bị cồng kềnh, tiêu thụ năng lượng lớn, khó tích hợp trên các nền tảng di động hoặc IoT.
- **Phổ tần (Spectrum):** Lãng phí tài nguyên vô tuyến khi phải phân mảnh băng tần cho từng chức năng riêng biệt.

6.1.2 Giải pháp JCAS (Joint Communication and Sensing)

JCAS đề xuất tích hợp cả hai chức năng vào một nền tảng phần cứng duy nhất, sử dụng chung một dạng sóng tín hiệu.

Nguyên lý hoạt động: Một xung sóng duy nhất thực hiện song song hai nhiệm vụ:

1. Mang dữ liệu thông tin tới người dùng (Downlink communication).
2. Thu nhận tín hiệu phản xạ từ môi trường để ước lượng tham số vật thể (Radar sensing).

Ưu điểm vượt trội:

- Giảm 50% phần cứng anten và bộ xử lý RF.
- Tối ưu hóa hiệu quả sử dụng phổ tần.
- Loại bỏ nhiễu giao thoa chéo giữa các hệ thống.

6.1.3 Bài toán Trade-off (Cân bằng)

Việc dùng chung tài nguyên dẫn đến bài toán đánh đổi hiệu năng:

- Tập trung năng lượng cho Communication $\rightarrow$ Giảm vùng phủ và độ nhạy Sensing.
- Phân tán năng lượng quét Sensing $\rightarrow$ Giảm SNR và throughput của Communication.

Câu hỏi nghiên cứu: Tối ưu hóa phân bổ năng lượng và trọng số anten để đạt điểm cân bằng Pareto giữa hai chức năng.

6.2 Thiết lập Mục tiêu Tối ưu

Mục tiêu là xác định vector trọng số Beamforming $\mathbf{W} \in \mathbb{C}^{12}$ cho mảng 12 anten phát.

6.2.1 Hàm mục tiêu (Objective Function)

Cực tiểu hóa sai số bình phương giữa pattern bức xạ thực tế và pattern mong muốn trên toàn bộ không gian góc:

$$\boxed{\min_{\mathbf{W}\in\mathbb{C}^{12}} \quad \mathcal{J}(\mathbf{W}) = \sum_{i=1}^{Q} \left|\mathbf{W}^H \mathbf{A}(\theta_i) - P_d(\theta_i)\right|^2} \tag{24}$$

Trong đó:

W: Vector trọng số cần tìm.

$\mathbf{A}(\theta_i)$: Vector đáp ứng mảng (Steering vector) tại góc θ_i.

$P_d(\theta_i)$: Giá trị pattern mong muốn tại góc θ_i.

Q: Số lượng điểm lấy mẫu góc (thường $Q = 320$ từ -90° đến $+90^\circ$).

6.2.2 Cấu trúc Pattern mong muốn (P_d)

Pattern lý tưởng được thiết kế dựa trên yêu cầu hệ thống:

- **Main lobe (Communication):** Tập trung tại 0°, độ lợi cao nhất.
- **Side lobes (Sensing):** Phân bố tại $\pm 30^\circ$, độ lợi thấp hơn.
- **Suppression region:** Các vùng còn lại yêu cầu mức Sidelobe < -20 dB.

6.2.3 Mô hình Hybrid Beamforming

Để điều khiển linh hoạt sự ưu tiên giữa hai chức năng, trọng số JCAS được mô hình hóa dưới dạng tổ hợp tuyến tính:

$$\mathbf{W}_{\text{JCAS}} = \sqrt{\rho} \cdot \mathbf{W}_{\text{comm}} + \sqrt{1-\rho} \cdot \mathbf{W}_{\text{sense}} \tag{25}$$

Bảng tham số trọng số $\rho \in [0, 1]$:

Bảng 2: Ý nghĩa tham số ρ trong ứng dụng thực tế

ρ	**Tỷ lệ (Comm/Sense)**	**Kịch bản ứng dụng**
1.0	100% / 0%	Truyền dữ liệu thuần túy (như Wi-Fi)
0.7	70% / 30%	Streaming video kết hợp giám sát
0.5	50% / 50%	Chế độ cân bằng tiêu chuẩn
0.3	30% / 70%	Xe tự lái (ưu tiên phát hiện vật cản)
0.0	0% / 100%	Radar thuần túy

6.3 Mô hình Toán học Hệ thống

6.3.1 Biến quyết định (Vector W)

Vector trọng số bao gồm biên độ và pha cho mỗi phần tử phát:

$$\mathbf{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{12} \end{bmatrix}, \quad w_m = |w_m| e^{j\phi_m} \tag{26}$$

Ý nghĩa vật lý:

- $|w_m|$: Kiểm soát công suất phát của anten thứ m.
- ϕ_m: Kiểm soát độ trễ pha để lái búp sóng.

6.3.2 Vector đáp ứng mảng (Steering Vector)

Với mảng anten đều (ULA) gồm $M = 12$ phần tử, khoảng cách $d = \lambda/2$, vector đáp ứng tại góc θ là:

$$\mathbf{A}(\theta) = \begin{bmatrix} 1 \\ e^{j\pi \sin(\theta)} \\ e^{j2\pi \sin(\theta)} \\ \vdots \\ e^{j(M-1)\pi \sin(\theta)} \end{bmatrix} \tag{27}$$

Nguyên lý hình thành búp sóng dựa trên sự giao thoa:

- **Đồng pha:** Tín hiệu cộng hưởng tăng cường (Constructive interference).
- **Ngược pha:** Tín hiệu triệt tiêu lẫn nhau (Destructive interference).

6.3.3 Thuật toán Two-Step Iterative Least Squares (TS-ILS)

Do bài toán gốc (Phương trình 24) là không lồi (non-convex) vì chứa hàm trị tuyệt đối, thuật toán TS-ILS được áp dụng để xấp xỉ nghiệm tối ưu.

Quy trình lặp:

1. **Bước 1 (Inner Loop - Cố định pha):**

   Giả sử biết pha ϕ của pattern mong muốn, giải bài toán bình phương tối thiểu:

$$\min_{\mathbf{W}} \|\mathbf{A}^H\mathbf{W} - P_d \odot e^{j\phi}\|^2 \tag{28}$$

   Nghiệm tường minh:

$$\mathbf{W}_{\text{opt}} = (\mathbf{A}\mathbf{A}^H)^{-1}\mathbf{A}(P_d \odot e^{j\phi}) \tag{29}$$

2. **Bước 2 (Outer Loop - Cập nhật pha):**

   Tính toán lại pha dựa trên vector **W** vừa tìm được:

$$\phi^{\text{new}} = \angle(\mathbf{W}^H\mathbf{A}) \tag{30}$$

3. **Kiểm tra hội tụ:** Lặp lại cho đến khi sai số $< \epsilon$.

Độ phức tạp tính toán xấp xỉ $O(K \cdot M^3)$, với K là số vòng lặp và M là số anten.

6.4 Các Ràng buộc Vật lý và Thiết kế

Để đảm bảo tính khả thi trong triển khai thực tế, nghiệm **W** phải thỏa mãn tập hợp các ràng buộc sau:

6.4.1 Ràng buộc Công suất (Power Constraint)

Tổng công suất phát không được vượt quá giới hạn phần cứng:

$$\|\mathbf{W}\|^2 = \sum_{m=1}^{M} |w_m|^2 \leq P_{\text{max}} \tag{31}$$

Vi phạm điều kiện này dẫn đến bão hòa bộ khuếch đại công suất (PA) và méo tín hiệu phi tuyến.

6.4.2 Ràng buộc Sidelobe (SLL)

Để giảm thiểu nhiễu và bảo mật thông tin, các búp sóng phụ phải bị nén xuống mức thấp:

$$|P(\theta)| \leq -20 \text{ dB}, \quad \forall\theta \notin \Omega_{\text{target}} \tag{32}$$

Tương đương biên độ tuyến tính ≤ 0.1 so với búp chính.

6.4.3 Ràng buộc Nulls (Triệt tiêu nhiễu)

Đặt các điểm ”không” (nulls) tại các hướng nhạy cảm hoặc không mong muốn (ví dụ $\pm 90^\circ$):

$$|\mathbf{W}^H\mathbf{A}(\theta_{\text{null}})| \approx 0 \tag{33}$$

Điều này giúp tránh lãng phí năng lượng vào các vùng vô ích.

6.4.4 Ràng buộc Đa chùm sóng (Multi-beam Isolation)

Khi hệ thống phát đa chùm sóng (cho nhiều người dùng hoặc mục tiêu), cần đảm bảo sự cách ly:

$$\Delta(\text{Beam}_i, \text{Beam}_j) \geq \Delta_{\min} \tag{34}$$

Nhằm hạn chế nhiễu xuyên âm (crosstalk) giữa luồng dữ liệu liên lạc và luồng tín hiệu radar.

7 TRIỂN KHAI GWO CHO BÀI TOÁN JCAS BEAMFORMING

7.1 Ánh xạ Bài toán JCAS sang Không gian Tối ưu GWO

7.1.1 Mapping các thành phần

Để áp dụng Grey Wolf Optimization vào bài toán JCAS, ta thực hiện các ánh xạ sau:

Bảng 3: Ánh xạ bài toán JCAS sang thuật toán GWO

Bài toán JCAS Beamforming	**GWO Component**
Vector trọng số $\mathbf{W} \in \mathbb{C}^{12}$	Vị trí con sói $\mathbf{X}_i$
Beam pattern tốt nhất	Alpha wolf (α)
Pattern tốt thứ hai	Beta wolf (β)
Pattern tốt thứ ba	Delta wolf (δ)
Các ứng viên còn lại	Omega wolves (ω)
Hàm mục tiêu (Eq. 24)	Fitness function $f(\mathbf{X}_i)$

7.1.2 Không gian tìm kiếm Complex-valued

Điểm khác biệt quan trọng: Vector beamforming $\mathbf{W}$ là **số phức**, không phải số thực như GWO truyền thống. Mỗi phần tử:

$$w_m = a_m + jb_m, \quad a_m, b_m \in \mathbb{R} \tag{35}$$

Do đó, không gian tìm kiếm là $\mathbb{C}^{12} \equiv \mathbb{R}^{24}$ (12 phần thực + 12 phần ảo).

Khởi tạo quần thể:

$$\mathbf{W}_i(0) = (\text{rand}(1, 12) - 0.5) + j \cdot (\text{rand}(1, 12) - 0.5) \tag{36}$$

Với $\text{rand}(1, 12)$ sinh số ngẫu nhiên phân phối đều trong $[0, 1]$.

7.2 Cấu hình Thực nghiệm

7.2.1 Thiết lập Hyperparameters

Để đánh giá ảnh hưởng của hyperparameters, nhóm thiết kế ba kịch bản với các siêu tham số khác nhau:

Bảng 4: Ba kịch bản Hyperparameter Tuning cho GWO variants

Kịch bản	**Số sói** (N)	**Số vòng lặp** ($T_{\max}$)	**Mục đích**
Fast	20	100	Kiểm tra nhanh, prototyping
Balanced	30	300	Cân bằng tốc độ-chất lượng
Thorough	50	500	Chất lượng tối đa, benchmark

Giải thích lựa chọn hyperparameters:

- **Kịch bản Fast (20 wolves, 100 iterations):**
  - Tổng số fitness evaluations: $20 \times 100 = 2{,}000$
  - Thời gian chạy ước tính: 15-30 giây
  - Phù hợp cho: Kiểm tra nhanh code, debugging, prototyping
  - Trade-off: Có thể chưa hội tụ đủ, kết quả kém ổn định
- **Kịch bản Balanced (30 wolves, 300 iterations):**
  - Tổng số fitness evaluations: $30 \times 300 = 9{,}000$
  - Thời gian chạy ước tính: 1-2 phút
  - Phù hợp cho: Thí nghiệm chính, so sánh thuật toán
  - Trade-off: Cân bằng giữa exploration (30 wolves) và exploitation (300 iter)
- **Kịch bản Thorough (50 wolves, 500 iterations):**
  - Tổng số fitness evaluations: $50 \times 500 = 25{,}000$
  - Thời gian chạy ước tính: 3-5 phút
  - Phù hợp cho: Benchmarking cuối cùng, đảm bảo hội tụ
  - Trade-off: Chi phí tính toán cao nhất nhưng kết quả tin cậy nhất

Lý do thiết kế ba kịch bản:

1. **Trade-off exploration-exploitation:** Số sói nhiều ($N \uparrow$) tăng khả năng khám phá không gian, số vòng lặp nhiều ($T_{\max} \uparrow$) tăng khai thác nghiệm tốt.
2. **Computational budget:** Ba kịch bản tương ứng với ba mức ngân sách: Low (2K), Medium (9K), High (25K) fitness evaluations.
3. **Sensitivity analysis:** So sánh hiệu năng các thuật toán khi thay đổi hyperparameters giúp xác định thuật toán nào robust hơn.

Hyperparameters cố định cho tất cả kịch bản:

- **Hệ số giảm a:** Giảm tuyến tính từ 2 về 0 theo công thức $a = 2 - t \cdot (2/T_{\max})$
- **Chaotic map parameter:** $\mu = 4$ (cho Chaotic GWO)

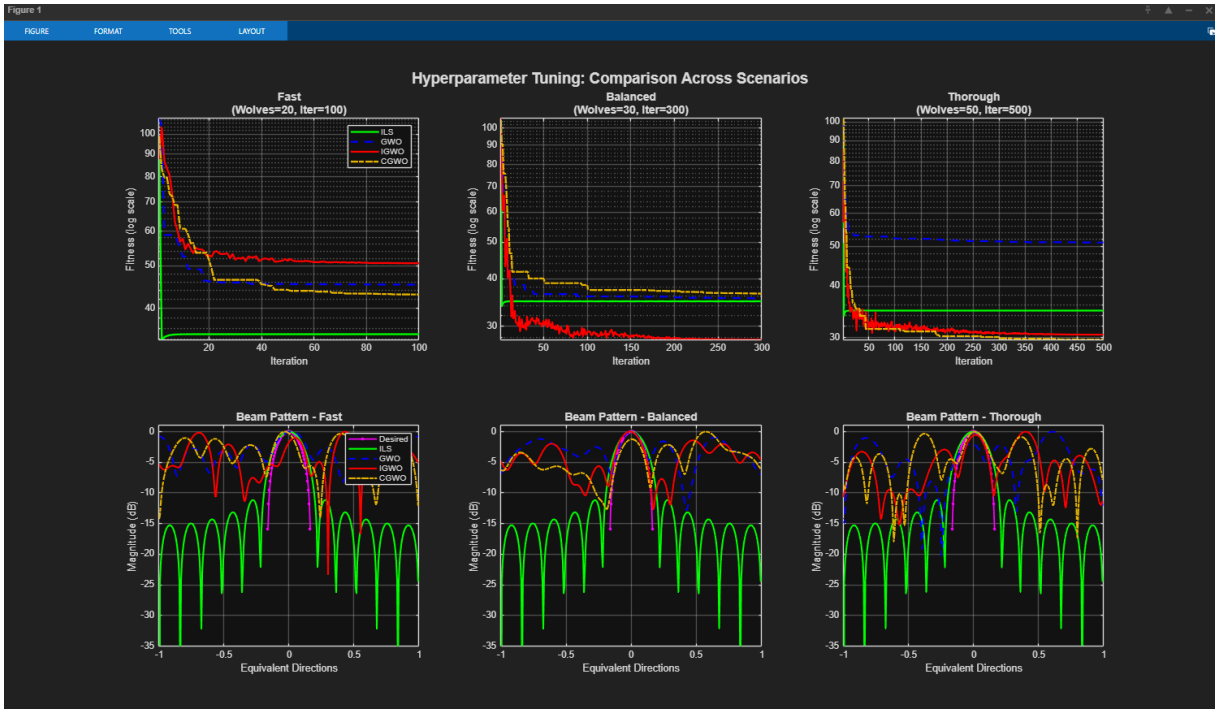
- **Initial chaotic value:** $Ch_0 = 0.7$
- **Random seed:** Không cố định để đảm bảo tính ngẫu nhiên thật sự
- **Convergence tolerance:** $\epsilon = 10^{-6}$ (dừng sớm nếu không cải thiện)

7.2.2 Thông số Hệ thống JCAS

- **Số phần tử anten:** $M = 12$ (ULA, khoảng cách $\lambda/2$).
- **Tần số sóng mang:** Giả định $\lambda = 1$ (chuẩn hóa).
- **Số điểm lấy mẫu góc:** $Q = 320$ từ $-90°$ đến $+90°$ (bước $0.1°$).
- **Hướng truyền thông:** $\theta_{\text{comm}} = 0°$ (Boresight).
- **Hướng cảm biến:** Quét từ $-80°$ đến $+80°$ (loại trừ main lobe).
- **Tham số cân bằng:** $\rho = 0.5$ (50% Communication - 50% Sensing).

8 KẾT QUẢ THỰC NGHIỆM

Hình 4 minh họa đường cong hội tụ fitness (hàng trên) và beam pattern tốt nhất (hàng dưới) của bốn thuật toán: TS-ILS, GWO, IGWO và CGWO trên ba kịch bản hyperparameter tuning.



Hình 4: So sánh hiệu năng các thuật toán trên ba kịch bản: Fast (20 sói, 100 iterations), Balanced (30 sói, 300 iterations), và Thorough (50 sói, 500 iterations). Hàng trên: đường cong hội tụ fitness. Hàng dưới: beam pattern của nghiệm tốt nhất.

8.1 So sánh Fitness theo Kịch bản

Bảng 5 tóm tắt thứ hạng fitness cuối cùng (fitness càng nhỏ càng tốt):

Bảng 5: Xếp hạng Fitness cuối cùng theo kịch bản (từ tốt nhất đến kém nhất)

extbfKịch bản	**Hạng 1**	**Hạng 2**	**Hạng 3**	**Hạng 4**
Fast (20 sói, 100 iter)	ILS	CGWO	GWO	IGWO
Balanced (30 sói, 300 iter)	IGWO	ILS	GWO	CGWO
Thorough (50 sói, 500 iter)	CGWO	IGWO	ILS	GWO

8.2 Nhận xét Kịch bản Thorough (50 sói, 500 iterations)

Quan sát cột phải của Hình 4:

- **Búp chính:** IGWO/CGWO khớp tốt với desired pattern quanh $\theta = 0$, trong khi ILS cho búp chính thấp hơn (kém khớp hơn).
- **Búp phụ:** ILS tạo các nulls sâu (búp phụ thấp hơn), còn IGWO/CGWO có búp phụ cao hơn nhưng đổi lại búp chính khớp tốt hơn.

Kết luận: Thứ hạng fitness phản ánh mức độ khớp tổng thể theo hàm mục tiêu đã chọn; vì vậy fitness thấp không nhất thiết đồng nghĩa với búp phụ thấp nhất. Nếu ưu tiên triệt búp phụ, cần thiết kế hàm mục tiêu có trọng số/penalty cho sidelobe.

Tài liệu

[1] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[2] R. Kumar *et al.*, "A novel hybrid pso-gwo approach for unit commitment problem," *Neural Computing and Applications*, vol. 27, pp. 1643–1655, 2022.

[3] T. Nguyen *et al.*, "A hybrid pso-gwo-based phase shift design for a hybrid-ris-aided heterogeneous network system," *Scientific Reports*, vol. 14, p. 1000, 2024.