DOI: https://doi.org/10.62985/j.huit_ojs.vol26.no1E.354

THUẬT TOÁN TỐI ƯU HÓA KẾT HỢP: KẾT HỢP TỐI ƯU HÓA SÓI XÁM VỚI THUẬT TOÁN TÌM KIẾM CHIM HỒNG HẠC

Đinh Nguyễn Trọng Nghĩa, Nguyễn Thị Thu Tâm*

Trường Đại học Công Thương Thành phố Hồ Chí Minh

*Email: *tamntt@huit.edu.vn*

Ngày nhận bài: 12/3/2024; Ngày nhận bài sửa: 12/5/2024; Ngày chấp nhận đăng: 31/5/2024

TÓM TẮT

Bài báo này trình bày một phương pháp tối ưu mới bằng cách kết hợp hai thuật toán metaheuristics mạnh mẽ: Thuật toán Tìm kiếm Flamingo (FSA) và thuật toán Tối ưu hóa Sói xám (GWO). FSA lấy cảm hứng từ hành vi di cư và tìm kiếm thức ăn của chim hồng hạc, trong khi GWO mô phỏng cơ chế săn bắt của sói xám. Thuật toán kết hợp này (được gọi là FGWO), tận dụng sức mạnh của cả FSA và GWO để cải thiện tốc độ hội tụ và khả năng tìm kiếm toàn cục của chúng. FGWO sử dụng GWO để tìm kiếm các giải pháp toàn cục sau đó áp dụng FSA cải thiện chất lượng giải pháp. Hiệu suất của FGWO được đánh giá trên một số hàm thử nghiệm và so sánh với các thuật toán tiên tiến khác. Kết quả thử nghiệm cho thấy FGWO vượt trội hơn các thuật toán khác về chất lượng giải pháp, độ ổn định và hiệu quả. Thuật toán kết hợp mới này góp phần vào sự phát triển của các phương pháp tối ưu hóa, cung cấp một góc nhìn mới về cách các thuật toán metaheuristics khác nhau có thể kết hợp để đạt được hiệu suất tốt hơn.

Keywords: Thuật toán tối ưu, thuật toán metaheuristic, tối ưu hóa sói xám, tối ưu hóa chim hồng hạc, tối ưu hóa kết hợp.

1. MỞ ĐẦU

Các thuật toán metaheuristic được ứng dụng để tìm kiếm các lời giải tối ưu gần đúng cho các bài toán tối ưu khó, trong đó không có một phương pháp chính xác nào có thể giải được trong thời gian hợp lý. Theo định nghĩa, thuật toán metaheuristic là một thuật toán không phụ thuộc vào bài toán mà có thể tìm ra các lời giải gần đúng cho các bài toán khó. Các thuật toán metaheuristic thường lấy cảm hứng từ tự nhiên và cố gắng giải bài toán bằng cách mô phỏng các hiện tượng vật lý, sinh học, v.v.[1]

Các thuật toán tiến hóa (EA) là những thuật toán metaheuristic dựa trên quần thể và có tính ngẫu nhiên. EA khác với các thuật toán thông thường khác bởi chúng thực hiện tiến hóa quần thể các lời giải để tiến đến một giải pháp tối ưu gần đúng. Thông thường, các EA thực hiện tìm kiếm trong không gian bài toán bằng cách phát sinh một quần thể các cá thể (lời giải) một cách ngẫu nhiên ban đầu, sau đó đánh giá độ thích nghi của chúng dựa trên hàm thích nghi. Trong các thế hệ tiếp theo, các cá thể tiến hóa theo xu hướng của cá thể tốt nhất và được hỗ trợ bởi hàm thích nghi. Quá trình này được tiếp tục cho đến khi đạt được số vòng lặp tối đa hoặc tìm được giải pháp gần đúng như mong đợi.

Dưới đây là một số thuật toán tiến hóa phổ biến:

- Thuật toán di truyền (Genetic Algorithm - GA): là một trong những thuật toán tiến hóa phổ biến nhất, sử dụng quy trình giải mã của di truyền học để tìm kiếm và tối ưu hóa các lời giải [2].
- Thuật toán tiến hóa đa nhiệm (Multi-Objective Evolutionary Algorithm - MOEA): được sử dụng để giải quyết các bài toán tối ưu đa mục tiêu, trong đó có nhiều hơn một hàm mục tiêu cần tối ưu [3].
- Thuật toán tối ưu hóa mạng nơ-ron (Neuro-evolution): sử dụng thuật toán tiến hóa để tối ưu hóa các mô hình mạng nơ-ron [4].
- Thuật toán lập trình di truyền (Genetic Programming - GP): sử dụng kỹ thuật di truyền để tạo ra các chương trình hoặc hàm tối ưu [5].

- Thuật toán tiến hóa phân bố (Differential Evolution - DE): sử dụng các phép toán số học để tạo ra các bản sao của các lời giải hiện tại và áp dụng các quy tắc tiến hóa để tìm kiếm lời giải tốt hơn. [6]

Trong bối cảnh đó, bài báo này đề xuất một phương pháp kết hợp độc đáo giữa thuật toán Grey Wolf Optimizer (GWO) [7-8] và Flamingo Search Algorithm (FSA) [9], mà chúng tôi gọi là Flamingo Grey-Wolf Optimizer (FGWO), nhằm tối ưu hóa hàm mục tiêu. GWO được lấy cảm hứng từ cách hoạt động của bầy sói xám trong tự nhiên, trong khi FSA lấy cảm hứng từ cách chim hồng hạc tìm kiếm thức ăn. Hai thuật toán này đã được chứng minh là hiệu quả trong nhiều bài toán tối ưu hóa, và việc kết hợp chúng có thể tạo ra một phương pháp mạnh mẽ và linh hoạt để giải quyết các bài toán phức tạp.

Phương pháp kết hợp GWO và FSA được đề xuất trong bài báo này tận dụng sự hội tụ nhanh của GWO và khả năng tìm kiếm cục bộ mạnh mẽ của FSA. Quy trình tối ưu hóa được chia thành hai giai đoạn, trong đó GWO được áp dụng trong giai đoạn đầu để khám phá và tìm kiếm toàn cục, trong khi FSA được triển khai trong giai đoạn thứ hai để cải thiện và điều chỉnh kết quả tìm kiếm, tập trung vào tìm kiếm cục bộ và tối ưu hóa kỹ thuật.

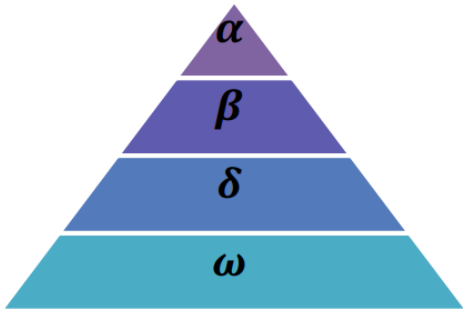
Với sự kết hợp này, phương pháp FGWO mang lại nhiều ưu điểm quan trọng. Nó không chỉ tạo ra một quá trình tìm kiếm mạnh mẽ và hiệu quả, mà còn mang lại tính linh hoạt và đa dạng trong quá trình tìm kiếm. Đồng thời, nó cũng có khả năng tối ưu hóa đa mục tiêu, mở ra cánh cửa cho tối ưu hóa hiệu quả của các bài toán đa mục tiêu phức tạp.

Bài báo này đề xuất phương pháp FGWO và kiểm chứng hiệu quả thông qua thực nghiệm trên các hàm thử nghiệm phổ biến. Kết quả cho thấy rằng FGWO có khả năng tìm ra giá trị tối ưu tốt và hội tụ nhanh chóng.

2. THUẬT TOÁN TỐI ƯU HÓA SÓI XÁM

2.1. Ý tưởng thuật toán

Thuật toán sói xám lấy cảm hứng từ hành vi xã hội của loài sói, hệ thống phân cấp xã hội của đàn sói xám. Theo đó trong bầy sói xám thông thường được phân cấp thành bốn nhóm được gọi tên theo thứ tự lần lượt từ cao đến thấp Alpha (α), Beta (β), Delta (δ) và Omega (ω) như thể hiện ở Hình 1. Mỗi vai trò trong bầy sói xám đảm bảo một nhiệm vụ khác nhau đảm bảo cho sự phát triển và tồn tại của bầy sói. Trong đó Alpha (α) là con đầu đàn và là con sói chiếm ưu thế nhất, Alpha (α) có thể là con đực hoặc con cái, chúng có nhiệm vụ đưa ra những quyết định về chiến lược săn mồi, tìm chỗ ở và phân chia thức ăn. Nhóm thứ hai là Beta (β), chúng có nhiệm vụ hỗ trợ cho Alpha (α) đưa ra quyết định. Những nhóm Beta (β) này có khả năng trở thành nhóm Alpha (α) trong trường hợp các con trong nhóm Alpha (α) rời đi hoặc chết đi chúng chiếm ưu thế cao hơn so với Delta (δ). Nhiệm vụ của những con sói này là đào tạo, lính canh gác và đảm bảo sự an toàn của bầy đàn. Nhóm cuối cùng có vị thế thấp nhất trong đàn là Omega (ω), đóng vai trò là con phải hy sinh, phải tuân theo những yêu cầu và mệnh lệnh của những con sói (α, β, δ) trong đàn.



Hình 1. Phân cấp của bầy sói trong tự nhiên [7]

Bên cạnh thứ bậc xã hội, sói xám có cách săn mồi rất đặc trưng với chiến lược độc đáo. Chúng đi săn theo bầy và hợp tác thành nhóm để tách con mồi ra khỏi đàn, sau đó một hoặc hai con sói sẽ đuổi theo và tấn công con mồi trong khi những con khác đuổi theo bất kỳ kẻ nào đi lạc.

Chiến lược săn mồi của bầy sói bao gồm:

- Tiếp cận, theo dõi và đuổi theo con mồi.
- Truy đuổi, quấy rối và bao vây xung quanh con mồi cho đến khi nó ngừng di chuyển.
- Tấn công con mồi khi nó kiệt sức.

2.2. Mô hình toán học và thuật toán

Phần này trình bày mô hình toán học và giới thiệu về thuật toán GWO. Các công thức liên quan đến các hệ thống xã hội, theo dõi, bao vây và tấn công con mồi sẽ được trình bày chi tiết.

2.2.1. Hệ thống phân cấp xã hội

Trong GWO, các con sói alpha (α) ứng với các giải pháp phù hợp nhất (lời giải tốt nhất của bài toán tại thời điểm hiện tại), hai giải pháp phù hợp tiếp theo được đặt tên lần lượt là beta (β) và delta (δ). Các giải pháp ứng viên còn lại được giả định là omega (ω), cũng được gọi là các tác nhân tìm kiếm. Đây chính là hệ thống phân cấp trong GWO. Do đó, việc săn mồi (tối ưu hóa) được định hướng bởi α, β và δ. Những con sói ω tuân thủ theo ba con sói này.

2.2.2. Bao vây con mồi

Mô hình toán học các bước bao vây trong GWO được biểu thị bởi các phương trình sau đây [7]:

$$\overrightarrow{D} = \left| \overrightarrow{C} \times \overrightarrow{X_p}(t) - \overrightarrow{X}(t) \right| \quad (1)$$

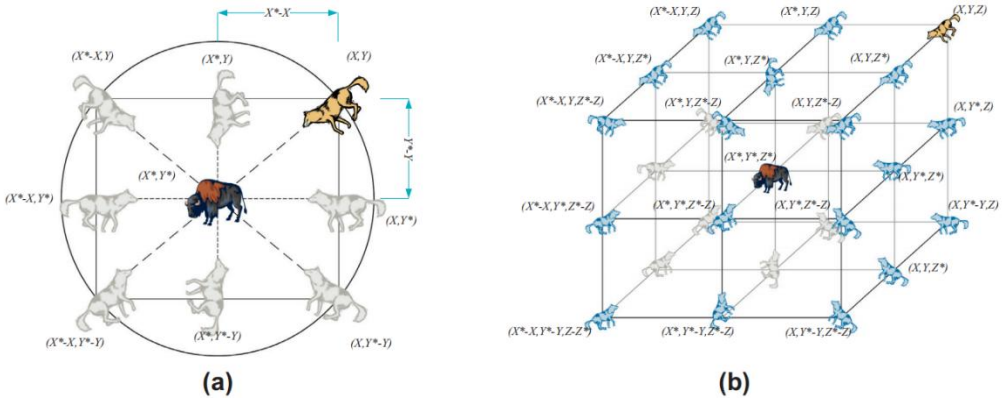
$$\overrightarrow{X}(t+1) = \overrightarrow{X_p}(t) - \overrightarrow{A} \times \overrightarrow{D} \quad (2)$$

trong đó t xác định thời điểm hiện tại (bước lặp), $\overrightarrow{A}$ và $\overrightarrow{C}$ là các vector hệ số, $\overrightarrow{X_p}$ là vector vị trí của con mồi, và $\overrightarrow{X}$ biểu thị vector vị trí của một con sói xám. Với

$$\overrightarrow{A} = 2 \times \overrightarrow{a} \times \overrightarrow{r_1} - \overrightarrow{a} \quad (3)$$

$$\overrightarrow{C} = 2 \times \overrightarrow{r_2} \quad (4)$$

trong đó các thành phần của $\overrightarrow{a}$ sẽ được giảm tuyến tính từ 2 xuống 0 trong quá trình lặp lại và r_1, r_2 là các vector ngẫu nhiên trong [0, 1]. Do đó, một con sói xám có thể cập nhật vị trí của nó theo tọa độ (X, Y) trong không gian xung quanh con mồi ở bất kỳ vị trí ngẫu nhiên nào bằng cách sử dụng các phương trình **(**1**)** và **(**4**)** thể hiện như Hình 2.



Hình 2. Mô phỏng vector vị trí 2D và 3D và các vị trí tiếp theo của sói xám

2.2.3. Săn mồi

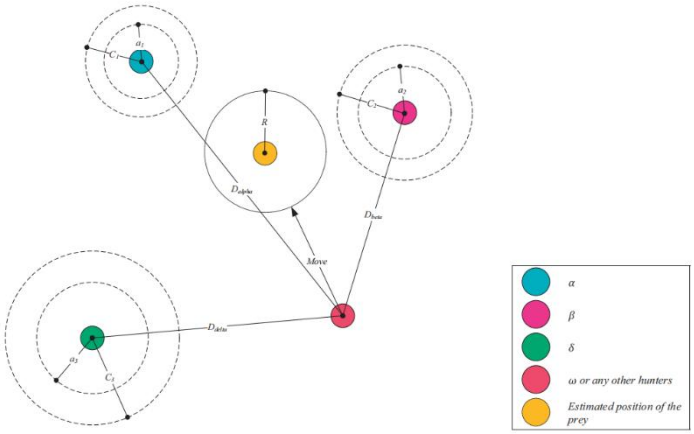
Sói xám có khả năng nhận biết vị trí của con mồi và tập trung bao vây chúng trong quá trình săn. Trong quá trình tối ưu hóa, alpha (α) thường được sử dụng để định hướng cho các giải pháp omega (ω). Tuy nhiên, khi không có thông tin về vị trí tối ưu (con mồi) trong không gian tìm kiếm, beta (β) và delta (δ) đồng thời được sử dụng. Vì vậy, thuật toán GWO lưu trữ ba giải pháp tốt nhất (bao gồm alpha (α), beta (β), delta (δ)), và yêu cầu các omega (ω) cập nhật vị trí theo vị trí của các giải pháp tốt nhất. Công thức được đề xuất cho vấn đề này như sau.

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} \times \overrightarrow{X_\alpha} - \overrightarrow{X}\right|, \overrightarrow{D_\beta} = \left|\overrightarrow{C_2} \times \overrightarrow{X_\beta} - \overrightarrow{X}\right|, \overrightarrow{D_\delta} = \left|\overrightarrow{C_3} \times \overrightarrow{X_\beta} - \overrightarrow{X}\right| \quad (5)$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1} \times (\overrightarrow{D_\alpha}), \overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2} \times (\overrightarrow{D_\beta}), \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} \times (\overrightarrow{D_\delta}) \quad (6)$$

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \quad (7)$$

Với các phương trình trên, các tác nhân tìm kiếm sẽ cập nhật vị trí theo vị trí của alpha, beta và delta trong không gian tìm kiếm. Kết quả cuối cùng sẽ được định vị trong một vòng tròn xung quanh alpha (α), beta (β) và delta (δ). Nói cách khác, alpha (α), beta (β) và delta (δ) sẽ xác định vị trí của con mồi và các tác nhân tìm kiếm sẽ cập nhật vị trí của mình xung quanh con mồi theo cách ngẫu nhiên như Hình 3.



Hình 3. Mô phỏng quá trình cập nhật vị trí của sói xám trong GWO

2.2.4. Tấn công con mồi (khai thác)

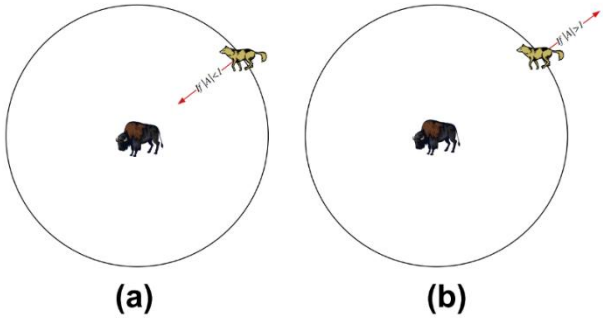
Sói xám kết thúc cuộc săn bằng cách tấn công con mồi khi con mồi ngừng di chuyển. Để lập mô hình toán học tiếp cận con mồi, thuật toán giảm giá trị của $\overrightarrow{a}$. Lưu ý rằng biên độ dao động của $\overrightarrow{A}$ cũng giảm theo $\overrightarrow{a}$. Nói cách khác, các phần tử của $\overrightarrow{A}$ sẽ là một giá trị ngẫu nhiên trong khoảng [-2a, 2a] trong đó a giảm từ 2 xuống 0 một cách tuyến tính qua quá trình lặp. Khi các phần tử của $\overrightarrow{A}$ nằm trong [-1, 1], vị trí tiếp theo của các tác nhân tìm kiếm có thể ở bất kỳ vị trí nào giữa vị trí hiện tại của nó và vị trí của con mồi. Hình 4 cho thấy khi |A| < 1, đàn sói tấn công về phía con mồi.

2.2.5. Tìm kiếm con mồi (thăm dò)

Trong thuật toán GWO, sói xám chủ yếu tìm kiếm con mồi theo vị trí của alpha (α), beta (β) và delta (δ). Chúng tách ra khỏi nhau để tìm kiếm con mồi và hội tụ để tấn công con mồi, tương tự như trong tự nhiên. Trong trường hợp phân kỳ (lời giải không hội tụ), các giá trị của $\overrightarrow{A}$ lớn hơn 1 hoặc nhỏ hơn -1 được sử dụng để bắt buộc các tác nhân tìm kiếm phải chuyển hướng khỏi con mồi, tạo ra sự đa dạng trong quá trình tìm kiếm và khám phá toàn diện.

Hình 4 cho thấy khi $|\overrightarrow{A}|$ >1 buộc những con sói phải tách khỏi con mồi để hy vọng tìm kiếm con mồi khác tốt hơn. Thành phần khác của GWO là $\overrightarrow{C}$, chứa các giá trị ngẫu nhiên trong [0, 2], cung cấp các trọng số ngẫu nhiên cho con mồi để xác định khoảng cách. Thành phần này hỗ trợ quá trình khám phá và tránh tối ưu hóa cục bộ, bằng cách tạo ra nhiều hành vi ngẫu nhiên hơn trong quá trình tối ưu hóa. Các chướng ngại vật trong tự nhiên cũng có thể được xem như tác động của $\overrightarrow{C}$ đối với việc tiếp cận con mồi.

Quá trình tìm kiếm bắt đầu bằng việc tạo ra một quần thể sói xám ngẫu nhiên, sau đó các sói alpha (α), beta (β) và delta (δ) ước tính vị trí có thể xảy ra của con mồi và cập nhật khoảng cách của mỗi giải pháp ứng cử viên với con mồi. Tham số được giảm để tăng hoạt động thăm dò và khai thác tương ứng. Các giải pháp ứng cử viên có xu hướng khác với con mồi khi $\overrightarrow{|A|}$ > 1 và hội tụ về phía con mồi khi $\overrightarrow{|A|}$ < 1. Cuối cùng, thuật toán GWO kết thúc khi thỏa mãn một tiêu chí cuối cùng.



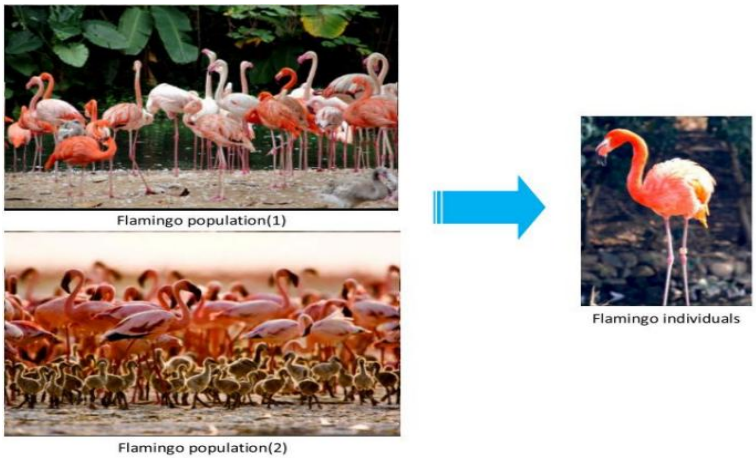
Hình 1. Mô phỏng quá trình tấn công con mồi so với tìm kiếm con mồi của sói xám

3. THUẬT TOÁN TÌM KIẾM CHIM HỒNG HẠC

3.1. Đặc điểm của thuật toán

Chim hồng hạc là loài chim di cư sống theo bầy đàn, thức ăn chủ yếu là tảo, tôm nhỏ, nghêu, giun nhỏ và ấu trùng côn trùng. Chim hồng hạc cũng kiếm ăn theo một cách đặc biệt, bằng cách cúi chiếc cổ dài xuống và quay đầu lại, sau đó đi bộ. Trong khi quét chiếc mỏ cong của chúng quanh cơ thể và chạm vào đáy nước để kiếm ăn [9]. Các quần thể và cá thể chim hồng hạc trong tự nhiên được thể hiện trong Hình 5.

Hai đặc điểm tập tính chính của chim hồng hạc là tập tính kiếm ăn và di cư. Quần thể chim hồng hạc chủ yếu sinh sống ở những khu vực có nhiều thức ăn. Sau một thời gian kiếm ăn rộng rãi, quần thể chim hồng hạc di cư khi thức ăn trong khu vực bị giảm đến mức không thể đáp ứng quần thể.



Hình 5. Quần thể và cá thể chim hồng hạc trong tự nhiên

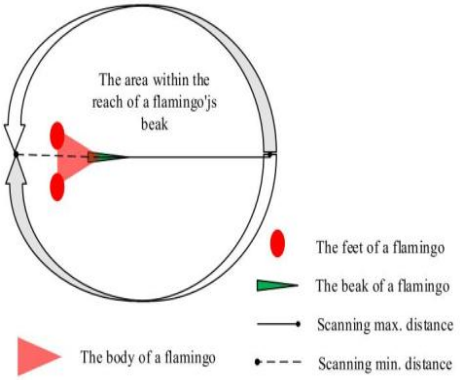
Các ý tưởng tối ưu hóa chính của mô hình FSA được trình bày như sau: Chim hồng hạc hát cho nhau nghe để thông báo vị trí của chúng và thức ăn tại địa điểm. Quần thể chim hồng hạc không biết nơi nào có nhiều thức ăn nhất trong khu vực tìm kiếm hiện tại. Thay vào đó, chúng cập nhật qua tiếng hát của đồng loại xung quanh. Hành vi này của hồng hạc phù hợp với ý tưởng tối ưu hóa trí thông minh bầy đàn. Và chúng ta không biết không gian tìm kiếm là bao nhiêu, tác nhân tìm kiếm là chim hồng hạc, hồng hạc khám phá ra không gian tìm kiếm và trao đổi thông tin với nhau và cố định vị trí tìm thấy đó. Từ đó rút ra được giải pháp tối ưu. Quy tắc thay đổi vị trí dựa trên hành vi của chim hồng hạc. Hành vi chính của nó chủ yếu có hai loại: hành vi kiếm ăn và hành vi di cư. Hành vi kiếm ăn có thể được chia thành hai đặc điểm hành vi: Quét mỏ và di chuyển chân.

3.2. Ý tưởng và mô hình toán học của thuật toán

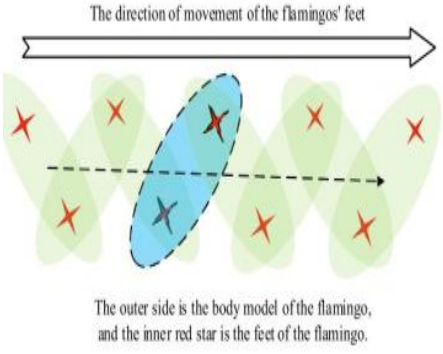
Tìm hiểu hành vi: Hành vi giao tiếp khi con hạc phát hiện chỗ có thức ăn nó sẽ gọi những con hạc khác đến vị trí đó. Chỗ tụ tập nhiều nhất là chỗ có nhiều thức ăn nhất. Theo lí thuyết những con hạc này

không biết chỗ nào có nhiều thức ăn. Nhưng trong thuật toán chúng ta có thể biết vì ta có thể thiết lập điều kiện kết thúc chương trình.

FSA là một thuật toán mô phỏng chim hồng hạc cố gắng tìm giải pháp tối ưu trong khu vực tìm kiếm (tức là vị trí có nhiều thức ăn nhất) dựa trên nguồn thông tin hạn chế có sẵn.



Hình 6. Hành vi quét mỏ



Hình 7. Hành vi di chuyển hai chân

Mỏ của chim hồng hạc khi úp ngược trong nước có chức năng tương tự như một cái sàng lớn, giúp hút nước và lọc thức ăn nhanh chóng nhờ vào các rãnh sâu ở mỏ dưới và các rãnh nông, có nắp ở mỏ trên. Khi chim hồng hạc đang kiếm ăn, chúng cúi đầu, lộn miệng ngược và ăn thức ăn vào miệng, sau đó thải ra nước thừa và cặn bã không ăn được. Cách kiếm ăn này phụ thuộc vào sự dồi dào thức ăn trong khu vực. Khi mỏ của chim hồng hạc quét qua một khu vực có nhiều thức ăn hơn, chúng sẽ quét khu vực đó một cách cẩn thận hơn và cổ của chim sẽ dần duỗi ra, làm tăng bán kính quét của mỏ. Xác suất quét khu vực để tìm thức ăn cũng sẽ tăng lên. Mô hình hành vi quét mỏ của chim hồng hạc được mô tả trong Hình 6.

Khi có nhiều con chim hồng hạc trong quần thể đồng thời hội tụ tại một vị trí, khả năng có nhiều thức ăn ở khu vực đó càng cao. Hành vi quét mỏ của chim hồng hạc được mô phỏng trong không gian đa chiều của quần thể được ký hiệu là x_{ij}, và mô hình tính đến sự thay đổi trong lựa chọn của từng cá thể chim hồng hạc trong tự nhiên cũng như tác động của sự biến động đột ngột của môi trường đến hành vi kiếm ăn của chúng. Hành vi tìm kiếm thức ăn của từng con chim hồng hạc sẽ có sai số khi truyền thông tin. Để mô phỏng sai số này, ta sử dụng một phân phối ngẫu nhiên theo phân phối chuẩn, trong đó việc quét mỏ của chim hồng hạc có xác suất cao hướng về vị trí có nhiều thức ăn nhất. Tuy nhiên, cũng có một xác suất nhỏ cho sai sót dựa trên thông tin này.

Khoảng cách tối đa mà mỏ chim hồng hạc quét qua trong quá trình tìm kiếm thức ăn được biểu diễn bằng công thức [9]: $|G_1 \times x_{ij} + \varepsilon_2 \times x_{ij}|$ trong đó ε_2 là một số ngẫu nhiên có giá trị -1 hoặc 1. Mục đích chính của khoảng cách tối đa này là mở rộng phạm vi tìm kiếm của mỏ chim hồng hạc trong quá trình tìm thức ăn. Trong đó, G_1 là một số ngẫu nhiên tuân theo phân phối chuẩn thông thường $N(0,1)$. Phạm vi quét mỏ của chim hồng hạc trong hành vi tìm kiếm được gọi là quét mỏ lần hai, và được biểu diễn bằng công thức [9]: $G2 \times |G1 \times xb_j + \varepsilon_2 \times x_{ij}|$ trong đó $G2$ là một số ngẫu nhiên tuân theo phân phối chuẩn $N(0,1)$.

Mô hình hành vi di chuyển bằng hai chân của hồng hạc được thể hiện trong Hình 7. Khi hồng hạc kiếm thức ăn, trong lúc quét bằng mỏ, móng vuốt của chúng sẽ di chuyển về phía nơi có nhiều thức ăn nhất. Nói một cách dễ hiểu là chúng vừa di chuyển vừa quét mỏ. Giả sử vị trí nơi thức ăn phong phú nhất trong quần thể là xb_j, khoảng cách đã đi được đo bằng "$\varepsilon_1 \times xb_j$, trong đó "$\varepsilon_1$ là một số ngẫu nhiên -1 hoặc 1, chủ yếu là để tăng phạm vi tìm kiếm của hồng hạc và định lượng sự khác biệt của cá thể trong việc lựa chọn.

Tổng kết lại, bước di chuyển của chim hồng hạc trong quá trình tìm kiếm thức ăn ở vòng lặp thứ t là tổng khoảng quét mỏ chim hồng hạc cộng với khoảng cách di chuyển của hai chân, như được hiển thị trong (8).

$$b_{ij}^t = \varepsilon_1 \times xb_j^t + G_2 \times |G_1 \times xb_j^t + \varepsilon_2 \times x_{ij}^t| \quad (8)$$

Công thức cập nhật vị trí kiếm ăn của chim hồng hạc được thể hiện như sau:

$$x_{ij}^{t+1} = (x_{ij}^{t} + \varepsilon_1 \times xb_j^t + G_2 \times |G_1 \times xb_j^t + \varepsilon_2 \times x_{ij}^t|)/K \quad (9)$$

Trong (9), x_{ij}^{t+1} đại diện vị trí của con hồng hạc thứ i trong chiều thứ j của quần thể trong lần lặp thứ t+1. x_{ij}^{t} đại diện vị trí của con hồng hạc thứ i trong chiều thứ j trong lần lặp thứ t của quần thể chim hồng hạc cụ thể là vị trí bàn chân chim hồng hạc xb_j^t: đại diện cho vị trí chiều thứ j của chim hồng hạc có độ thích nghi tốt nhất trong quần thể ở lần di cư. $K = K(n)$: là hệ số khuếch tán, là một số ngẫu nhiên tuân theo phân phối chuẩn. Nó được sử dụng để tăng quy mô phạm vi kiếm ăn của chim hồng hạc và để mô phỏng cơ hội lựa chọn cá thể trong tự nhiên, tăng khả năng tìm kiếm toàn cục của nó $G_1 = G_2 = N(0,1)$ là một số ngẫu nhiên tuân theo phân phối chuẩn. ε_1 và ε_2 là một số ngẫu nhiên -1 hoặc 1.

Khi thức ăn ít trong khu vực tìm kiếm hiện tại, quần thể hồng hạc di cư đến khu vực tiếp theo có thức ăn phong phú hơn. Giả sử vị trí của khu vực giàu thức ăn trong chiều thứ j là xb_j, công thức cho sự di cư của quần thể hồng hạc như sau:

$$x_{ij}^{t+1} = x_{ij}^{t} + \omega \times (xb_j^t - x_{ij}^t) \quad (10)$$

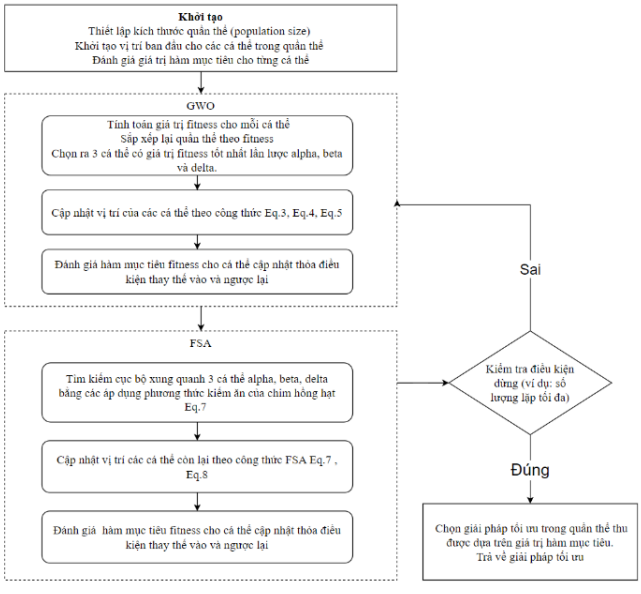
Trong (10), x_{ij}^{t+1} đại diện vị trí của con hồng hạc thứ i trong chiều thứ j của quần thể trong lần lặp thứ t+1. x_{ij}^{t} đại diện vị trí của con hồng hạc thứ i trong chiều thứ j trong lần lặp thứ t của quần thể chim hồng hạc cụ thể là vị trí bàn chân chim hồng hạc xb_j^t: đại diện cho vị trí chiều thứ j của chim hồng hạc có độ thích nghi tốt nhất trong quần thể ở lần di cư. $\omega = N(0,n)$: là một số ngẫu nhiên Gaussian với n bậc tự do, được sử dụng để tăng không gian tìm kiếm trong quá trình di cư của chim hồng hạc và mô phỏng tính ngẫu nhiên của các hành vi riêng lẻ của chim hồng hạc trong quá trình di cư cụ thể.

4. THUẬT TOÁN KẾT HỢP FLAMINGO GREY WOLF OPTIMIZATION (FGWO)

4.1. Ý tưởng

Ý tưởng chính của phương pháp kết hợp này là tận dụng sự hội tụ nhanh của Grey Wolf Optimizer (GWO) và khả năng tìm kiếm cục bộ mạnh mẽ của Flamingo Search (FSA). Quy trình tối ưu hóa được chia thành hai giai đoạn, trong đó GWO được áp dụng trong giai đoạn đầu để khám phá và tìm kiếm toàn cục, trong khi FSA được triển khai trong giai đoạn thứ hai để cải thiện và điều chỉnh kết quả tìm kiếm, tập trung vào tìm kiếm cục bộ và tối ưu hóa kỹ thuật.

Thuật toán FGWO có thể được viết dưới dạng lưu đồ sau



Hình 8. Lưu đồ thuật toán FGWO

4.2. Tối ưu hóa hàm toán học sử dụng thuật toán FGWO

Bài toán tối ưu hóa hàm nói chung có thể được áp dụng để xác định giá trị cực tiểu và cực đại của các hàm số trong một không gian tìm kiếm giới hạn. Trong bối cảnh đó, bài báo này trình bày sự áp dụng thuật toán FGWO để tối ưu hóa các hàm thử nghiệm được mô tả trong Bảng 1.

Bảng 1. Các hàm thử nghiệm được sử dụng

Tên hàm	Công thức toán học	Phạm vi
F1	$F_1(x) = \sum_{i=1}^{n} x_i^2$	[-2, 2]
F2	$F_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	
F3	$F3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	
F4	$F4(x) = max_i\{\lvert x_i \rvert, 1 \leq i \leq n\}$	

Các hàm F_n trong bảng 1 đều có cực tiểu toàn cục $F = 0$ khi $\vec{x} = 0$. Bài báo sử dụng các hàm thử nghiệm này để đánh giá khả năng tối ưu hóa của các thuật toán mục tiêu. Áp dụng thuật toán đề xuất FGWO lên các hàm này và đánh giá khả năng tìm kiếm điểm tối ưu (điểm cực tiểu) của thuật toán thông qua một số độ đo. Phạm vi của các hàm quy định giá trị ngẫu nhiên ban đầu được thiết lập, đồng thời cũng quy định điều kiện biên cho không gian tìm kiếm của bài toán.

Thuật toán FGWO để tối ưu hóa hàm mục tiêu có thể được triển khai theo các bước cụ thể sau:

- Bước 1: Khởi tạo quần thể ban đầu
  - ✓ Thiết lập số lượng cá thể và các tham số khác nhau cho GWO và FSA, như số lượng sói xám trong GWO và số lượng chim hồng hạc trong FSA.
  - ✓ Khởi tạo vị trí và tốc độ ban đầu của cá thể trong không gian tìm kiếm.
- Bước 2: Áp dụng GWO để tìm kiếm toàn cục
  - ✓ Tính toán giá trị hàm mục tiêu cho từng cá thể trong quần thể.
  - ✓ Xác định sói alpha, beta và delta trong GWO, đại diện cho các giải pháp tiềm năng tốt nhất trong quần thể.
  - ✓ Sử dụng công thức cập nhật GWO để di chuyển các cá thể trong quần thể theo sự hướng dẫn của sói alpha, beta và delta.
  - ✓ Tính toán lại giá trị hàm mục tiêu sau khi di chuyển.
- Bước 3: Áp dụng Flamingo Search (FSA) để điều chỉnh kết quả
  - ✓ Tính toán giá trị hàm mục tiêu cho từng cá thể trong quần thể sau giai đoạn GWO.
  - ✓ Áp dụng các phép biến đổi và tìm kiếm cục bộ trong FSA để cải thiện vị trí của các cá thể và giá trị hàm mục tiêu.
  - ✓ Cập nhật vị trí và tốc độ của các cá thể theo quy tắc của FSA.
  - ✓ Tính toán lại giá trị hàm mục tiêu sau khi di chuyển.
- Bước 4: Tiếp tục lặp lại giai đoạn GWO và FSA
  - ✓ Lặp lại giai đoạn GWO và FS theo số lần lặp được xác định trước đến khi điều kiện dừng được đáp ứng (ví dụ: số lần lặp tối đa, giá trị hàm mục tiêu đạt một ngưỡng xác định).
  - ✓ Trong mỗi lần lặp lại, áp dụng GWO để khám phá toàn cục và FSA để điều chỉnh kết quả.
- Bước 5: Đánh giá và chọn giải pháp tối ưu
  - ✓ Tính toán giá trị hàm mục tiêu cho tất cả các cá thể trong quần thể sau khi hoàn thành tất cả các lần lặp.
  - ✓ So sánh và lựa chọn giải pháp tốt nhất dựa trên giá trị hàm mục tiêu.
- Bước 6: Kết thúc quá trình tối ưu hóa
  - ✓ Trả về giải pháp tối ưu được chọn và giá trị hàm mục tiêu tương ứng.
  - ✓ Kết thúc quá trình tối ưu hóa.

5. KẾT QUẢ THỰC NGHIỆM

Trong bài báo này, tất cả các kết quả kiểm chứng được thực hiện trên máy tính có cấu hình AMD Ryzen 5 3550H, 16GB RAM và tốc độ xử lý 2.10 GHz, sử dụng phiên bản Python 3.11.2. Chúng tôi tiến hành đánh giá chất lượng của phương pháp kết hợp FGWO bằng cách so sánh với các thuật toán PSO [10], GOA [11], MFO [12], MPA [13], GWO [7], HHO [14], SSA [15], WOA [16]. Quá trình hội tụ và giá trị tối ưu nhất được tìm thấy cho các hàm thử nghiệm được mô tả trong Bảng 4.

Bảng 2. Giá trị khởi tạo ban đầu cho thuật toán

population_size	10
phạm vi	[-2,2]
dimension	3
MP_b (tỉ lệ di cư)	0.1

Bảng 2 mô tả các tham số điều khiển của thuật toán GWO và FSA. Vì các thuật toán trong thực nghiệm đều là thuật toán tối ưu ngẫu nhiên, thuật toán được lặp lại nhiều lần để thu thập thống kê các giá trị tìm kiếm của các trị tối ưu nhất và giá trị trung bình. Các hàm F_n sử dụng trong thực nghiệm được lấy từ Bảng 1. Dữ liệu đầu vào của các cá thể trong quần thể là các giá trị ngẫu nhiên trong khoảng [-2, 2] theo quy ước phạm vi của các hàm số này.

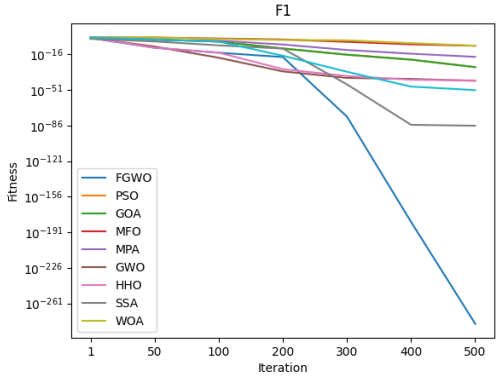
Điều này giúp đảm bảo tính chính xác và đáng tin cậy của các kết quả thu được từ phương pháp kết hợp FGWO.

Bảng 3. Thống kê trung bình kết quả thu được của các thuật toán

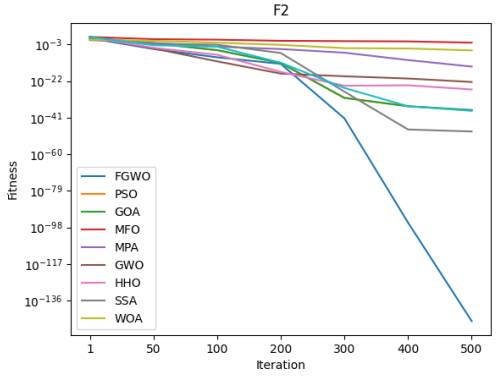
	F1	F2	F3	F4
PSO	8,64E-22	9,46E-36	4,86E-11	6,50E-08
GOA	1,03E-06	4,37E-02	8,96E-01	4,41E-03
MFO	6,34E-16	8,92E-12	3,76E+01	8,51E-02
MPA	9,44E-41	2,32E-21	4,90E-24	1,42E-17
GWO	1,33E-41	7,10E-25	5,19E-18	4,13E-12
HHO	6,61E-86	7,79E-48	7,55E-28	8,89E-38
SSA	1,58E-05	9,22E-06	8,86E-06	5,05E-04
WOA	2,76E-48	9,54E-36	2,67E+01	5,80E-02
FGWO	4,81E-281	3,14E-142	7,84E-251	1,39E-142

Bảng 4. Thống kê kết quả tốt nhất thu được của các thuật toán

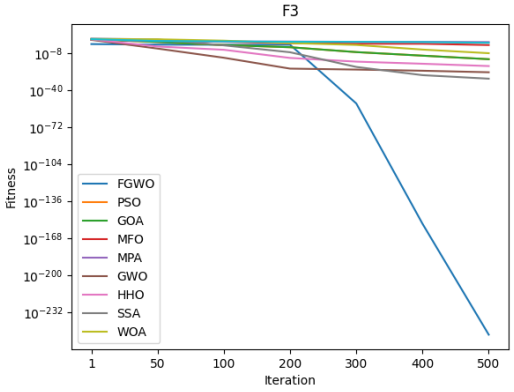
	F1	F2	F3	F4
PSO	3,86E-29	9,18E-38	4,42E-14	7,82E-09
GOA	3,99E-08	9,95E-03	7,29E-02	8,50E-04
MFO	4,85E-19	3,72E-15	2,70E+01	7,31E-02
MPA	1,27E-42	3,74E-23	2,37E-25	7,37E-18
GWO	2,94E-42	4,91E-27	5,30E-20	7,77E-13
HHO	7,93E-87	7,19E-49	6,85E-31	5,72E-39
SSA	4,39E-08	9,47E-07	6,55E-09	6,34E-06
WOA	8,76E-52	3,82E-38	5,83E+00	8,43E-03
FGWO	2,29E-291	1,98E-149	1,64E-260	3,64E-144



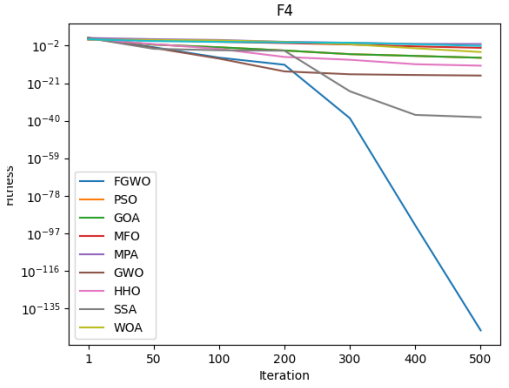
Hình 9. Biểu đồ thống kê kết quả thu được của các thuật toán ở hàm mục tiêu F1



Hình 10. Biểu đồ thống kê kết quả thu ở hàm mục tiêu F2



Hình 11. Biểu đồ thống kê kết quả thu ở hàm mục tiêu F3



Hình 12: Biểu đồ thống kê kết quả thu ở hàm mục tiêu F4

Qua việc so sánh với các thuật toán tối ưu khác như PSO, GOA, MFO, MPA, GWO, HHO, SSA, WOA ở Hình 9, Hình 10, Hình 11 và Hình 12 phương pháp FGWO đã đạt được quá trình hội tụ tốt hơn và tìm được giá trị tối ưu nhất cho các hàm thử nghiệm. Dựa trên các đồ thị, đối với các 4 hàm, bắt đầu khoảng vòng lặp thứ 200 là có sự hội tụ nhanh rõ rệt hơn các hàm khác. Ngoài ra, cho đến vòng lặp thứ 500 như trong các thực nghiệm, FGWO cho kết quả vượt trội hơn hẳn các hàm khác, thuật toán này đã tìm được kết quả tiệm cận nhất với cực tiểu toàn cục (~0) của các hàm thử nghiệm. Điều này cho thấy khả năng của FGWO trong tìm kiếm và khai thác không gian tìm kiếm một cách hiệu quả, đồng thời cung cấp kết quả tối ưu cao hơn so với các phương pháp khác. Thực nghiệm cũng đã xác nhận rằng việc kết hợp sự hội tụ nhanh của thuật toán Grey Wolf Optimizer (GWO) và khả năng tìm kiếm cục bộ mạnh mẽ của Flamingo Search (FSA) mang lại lợi ích đáng kể. Quy trình tối ưu hóa hai giai đoạn của phương pháp FGWO giúp tận dụng sự mạnh mẽ của cả hai thuật toán, bù trừ nhược điểm và tạo ra quá trình tìm kiếm mạnh mẽ và linh hoạt.

Kết quả tốt của phương pháp FGWO cũng được chứng minh thông qua việc thực nghiệm lặp lại quy trình nhiều lần. Các giá trị tìm kiếm của các trị tối ưu nhất và giá trị trung bình đã cung cấp sự tin cậy và chính xác cho phương pháp FGWO.

6. KẾT LUẬN

Mặc dù phương pháp tối ưu hóa FGWO đã chứng minh sức mạnh của mình trong việc tối ưu hóa không gian tìm kiếm, nhưng cũng cần nhấn mạnh rằng nó vẫn còn nhược điểm nhất định. Một trong những hạn chế đáng lưu ý là hiệu suất của FGWO chưa được chứng minh đối với các hàm tuần hoàn, nơi mà sự phức tạp của bài toán thường làm giảm hiệu suất của thuật toán.

Tuy nhiên, điều này không làm giảm đi giá trị của phương pháp này. Sự kết hợp giữa sự hội tụ nhanh của GWO và khả năng tìm kiếm cục bộ mạnh mẽ của FSA vẫn là một điểm mạnh đáng kể. Đặc biệt, trong các bài toán không gian tìm kiếm phức tạp, FGWO đã chứng tỏ khả năng tối ưu hóa hiệu quả và đáng kể.

Để tận dụng tối đa tiềm năng của FGWO, nghiên cứu tiếp theo có thể tập trung vào áp dụng phương pháp này vào các bài toán thực tế. Cụ thể, việc áp dụng FGWO cho các bài toán như lập lịch, tìm đường, hoặc các bài toán tối ưu hóa thực tế khác có thể mang lại những tiến bộ đáng kể và ứng dụng rộng rãi trong thực tế. Bằng cách này, FGWO không chỉ là một công cụ mạnh mẽ trong nghiên cứu khoa học mà còn có tiềm năng để giải quyết các vấn đề thực tiễn đang tồn tại trong các lĩnh vực khác nhau.

TÀI LIỆU THAM KHẢO

[1] I. Boussaïd, J. Lepagnot, and P. Siarry, "A survey on optimization metaheuristics," *Information Sciences*, vol. 237, pp. 82–117, 2013, doi: https://doi.org/10.1016/j.ins.2013.02.041.

[2] D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[3] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd ed. New York: Springer, 2007. doi: https://doi.org/10.1007/978-0-387-36797-2.

[4] J. R. Koza, M. A. Keane, J. Yu, et al., "Automatic creation of human-competitive programs and controllers by means of genetic programming," *Genetic Programming and Evolvable Machines*, vol. 1, pp. 121–164, 2000, doi: https://doi.org/10.1023/A:1010076532029.

[5] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002, doi: https://doi.org/10.1162/106365602320169811.

[6] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[7] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, doi: https://doi.org/10.1016/j.advengsoft.2013.12.007.

[8] X. Yu, W. Y. Xu, and C. L. Li, "Opposition-based learning grey wolf optimizer for global optimization," *Knowledge-Based Systems*, vol. 226, pp. 107–139, 2021, doi: https://doi.org/10.1016/j.knosys.2021.107139.

[9] W. Zhiheng and L. Jianhua, "Flamingo search algorithm: A new swarm intelligence optimization algorithm," *IEEE Access*, vol. 9, pp. 88564–88582, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3090512.

[10] C. Wang, L. Wang, and X. R. Xuejing, "General particle swarm optimization algorithm," in *Proc. 2023 IEEE 2nd Int. Conf. Electrical Engineering, Big Data and Algorithms (EEBDA)*, Changchun, China, 2023, pp. 1204–1208, doi: https://doi.org/10.1109/EEBDA56825.2023.10090725.

[11] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017, doi: https://doi.org/10.1016/j.advengsoft.2017.01.004.

[12] M. Bahrami, O. Bozorg-Haddad, and X. Chu, "Moth-flame optimization (MFO) algorithm," in *Advanced Optimization by Nature-Inspired Algorithms*, O. Bozorg-Haddad, Ed. Singapore: Springer, 2018.)

[13] Q. Fan, H. Huang, Q. Chen, et al., "A modified self-adaptive marine predators algorithm: Framework and engineering applications," *Engineering with Computers*, vol. 38, pp. 3269–3294, 2022, doi: https://doi.org/10.1007/s00366-021-01319-5.

[14] Z. M. Elgamal, N. B. M. Yasin, M. Tubishat, M. Alswaitti, and S. Mirjalili, "An improved Harris hawks optimization algorithm with simulated annealing for feature selection in the medical

field," *IEEE Access*, vol. 8, pp. 186638–186652, 2020, doi: https://doi.org/10.1109/ACCESS.2020.3029728.

[15] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017, doi: https://doi.org/10.1016/j.advengsoft.2017.07.002.

ABSTRACT

HYBRID METAHEURISTIC OPTIMIZATION: COMBINING GREY WOLF OPTIMIZATION WITH FLAMINGO SEARCH ALGORITHM

Dinh Nguyen Trong Nghia, Nguyen Thi Thu Tam*

Ho Chi Minh City University of Industry and Trade

*Email: *tamntt@huit.edu.vn*

This paper introduces a novel and innovative optimization method by combining two powerful metaheuristic algorithms: the Flamingo Search Algorithm (FSA) and the Grey Wolf Optimization (GWO) algorithm. FSA draws inspiration from the migration and foraging behavior of flamingos, while GWO simulates the hunting mechanism of grey wolves. The proposed hybrid algorithm, named FGWO, leverages the strengths of both FSA and GWO, capitalizing on their exploration and exploitation capabilities, convergence speed, and global search abilities. FGWO utilizes GWO for global solution search and subsequently employs FSA to enhance solution quality. The performance of FGWO is evaluated on a set of benchmark functions and compared against other state-of-the-art algorithms. Our experimental results demonstrate that FGWO outperforms other algorithms in terms of solution quality, stability, and efficiency. This novel hybrid algorithm contributes to the advancement of optimization methods, offering a fresh perspective on integrating diverse metaheuristic algorithms to achieve superior performance.

Keywords: Optimization Algorithm, Metaheuristic algorithm, Grey Wolf Optimizer, Flamingo Search Algorithm, Hybrid Optimization Algorithm.