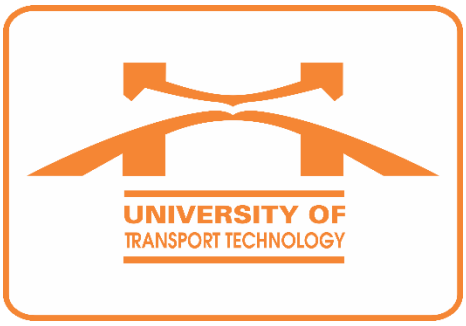
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ GTVT

KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO MÔN HỌC

TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI: GREY WOLF OPTIMIZER (GWO)

Sinh viên thực hiện: **Chu Đình Tạo**

Bùi Hữu Trung

Nguyễn Văn Phú

Nguyễn Minh Nghĩa

Nguyễn Quang Trường

Lớp: **73DCTT24**

Giảng viên hướng dẫn: Ths.Đỗ Bảo Sơn

Hà Nội 2024

MỤC LỤC

1. Giới thiệu 4

2. Grey Wolf Optimizer (GWO) 5

2.1 Cảm hứng 5

2.2. Mô hình toán học và thuật toán 7

3. Thực nghiệm 13

3.1 Vấn đề của GWO 13

3.2 Đột biến Cauchy-Gaussian 14

3.3 Thuật toán GG – GWO 15

3.4 Mô phỏng và kết quả 17

4. Kết luận 37

Bảng phân công công việc

STT	Mã SV	Thành viên	Công việc	Đóng góp	Ghi chú
1	73DCTT22423	Bùi Hữu Trung	ACO -	20%	
2	73DCTT23330	Chu Đình Tạo	ACO -	20%	
3	73DCTT23127	Nguyễn Quang Trường	GWO – Thực nghiệm, Demo thuật toán	20%	
4	73DCTT22213	Nguyễn Văn Phú	GWO – PowerPoint	20%	
5	73DCTT22255	Nguyễn Minh Nghĩa	GWO – Báo cáo	20%	

1. Giới thiệu

Các kỹ thuật tối ưu hóa siêu thuật toán đã trở nên rất phổ biến trong hai thập kỷ qua. Đáng ngạc nhiên là một số trong số chúng như Genetic Algorithm(Thuật toán di truyền) (GA), Ant Colony Optimization(Tối ưu hóa đàn kiến) (ACO) và Particle Swarm Optimization(Tối ưu hóa bầy hạt) (PSO) khá nổi tiếng không chỉ trong số các nhà khoa học máy tính mà còn trong số các nhà khoa học từ các lĩnh vực khác nhau. Bên cạnh số lượng lớn các công trình lý thuyết, các kỹ thuật tối ưu hóa như vậy đã được áp dụng trong nhiều lĩnh vực nghiên cứu khác nhau. Có một câu hỏi ở đây là tại sao siêu thuật toán lại trở nên phổ biến đáng kể. Câu trả lời cho câu hỏi này có thể được tóm tắt thành bốn lý do chính: tính đơn giản, tính linh hoạt, cơ chế không có đạo hàm và tránh tối ưu cục bộ.

The No Free Lunch(Không có bữa trưa miễn phí) (NFL) là định lý đáng được đề cập ở đây. Định lý này đã chứng minh một cách hợp lý rằng không có siêu thuật toán nào phù hợp nhất để giải quyết mọi vấn đề tối ưu hóa. Nói cách khác, một siêu thuật toán cụ thể có thể cho thấy kết quả rất hứa hẹn trên một tập hợp các vấn đề, nhưng cùng một thuật toán có thể cho thấy hiệu suất kém trên một tập hợp các vấn đề khác. Rõ ràng, NFL làm cho lĩnh vực nghiên cứu này trở nên rất năng động, dẫn đến việc tăng cường các phương pháp tiếp cận hiện tại và đề xuất các siêu thuật toán mới hàng năm. Điều này cũng thúc say các nỗ lực của chúng tôi nhằm phát triển một siêu thuật toán mới lấy cảm hứng từ loài sói xám.

Siêu thuật toán có thể được chia thành hai lớp chính: dựa trên giải pháp đơn và dựa trên quần thể. Một trong những nhánh thú vị của siêu thuật toán dựa trên quần thể là Swarm Intelligence(Trí tuệ bầy đàn) (SI). Các khái niệm về SI lần đầu tiên được đề xuất vào năm 1993 . Theo Bonabeau và các cộng sự, SI là “The emergent collective intelligence of groups of simple agents”(Trí thông minh tập thể nổi lên từ các nhóm tác nhân đơn giản). Những nguồn cảm hứng của các kỹ thuật trí tuệ bầy đàn (SI) chủ yếu bắt nguồn từ các quần thể tự nhiên, bầy chim, đàn thú, và đàn cá,... Một số kỹ thuật SI phổ biến nhất là ACO , PSO và and Artificial Bee Colony(Đàn ong nhân tạo) (ABC) . Một đánh giá toàn diện về các thuật toán SI được cung cấp trong phần tiếp theo. Một số lợi thế của các thuật toán SI là:

- Thuật toán SI bảo toàn thông tin về không gian tìm kiếm trong suốt quá trình lặp lại, trong khi Thuật toán tiến hóa (EA) loại bỏ thông tin của các thế hệ trước
- Các thuật toán SI thường sử dụng bộ nhớ để lưu giải pháp tốt nhất thu được cho đến nay
- Các thuật toán SI thường có ít tham số hơn để điều chỉnh

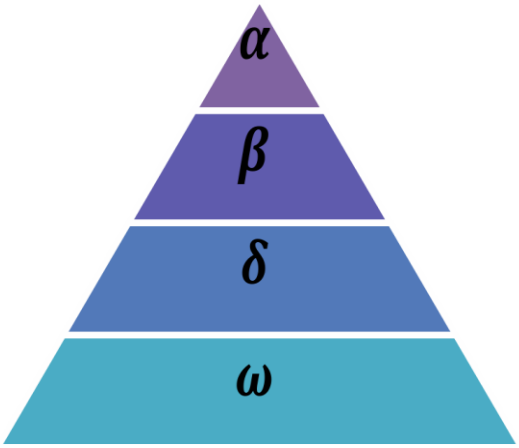
- Các thuật toán trí tuệ bầy đàn (SI) có ít toán tử hơn so với các phương pháp tiến hóa
- Thuật toán SI dễ dàng thực hiện

Bất kể sự khác biệt giữa các siêu thuật toán tìm kiếm, một đặc điểm chung là quá trình tìm kiếm được chia thành hai giai đoạn: thăm dò và khai thác. Giai đoạn thăm dò đề cập đến quá trình điều tra các khu vực có triển vọng của không gian tìm kiếm rộng nhất có thể. Một thuật toán cần có các toán tử ngẫu nhiên để tìm kiếm ngẫu nhiên và toàn cục trong không gian tìm kiếm nhằm hỗ trợ giai đoạn này. Tuy nhiên, khai thác đề cập đến khả năng tìm kiếm cục bộ xung quanh các khu vực có triển vọng thu được trong giai đoạn thăm dò. Việc tìm ra sự cân bằng thích hợp giữa hai giai đoạn này được coi là một nhiệm vụ đầy thách thức do bản chất ngẫu nhiên của siêu thuật toán tìm kiếm. Bài tìm hiểu này nghiên cứu về một kỹ thuật trí tuệ bầy đàn (SI) lấy cảm hứng từ hệ thống phân cấp xã hội và hành vi săn mồi của bầy sói xám. Như trước đó không có kỹ thuật SI nào trong tài liệu mô phỏng hệ thống phân cấp lãnh đạo của loài sói xám, nổi tiếng với hành vi săn mồi theo bầy đàn. Điều này thúc đẩy nghiên cứu nỗ lực mô hình hóa hành vi xã hội của loài sói xám theo phương pháp toán học, đề xuất một thuật toán SI mới lấy cảm hứng từ loài sói xám và nghiên cứu khả năng của nó trong việc giải quyết các vấn đề chuẩn mực và thực tế.

2. Grey Wolf Optimizer (GWO)

2.1 Cảm hứng

Sói xám (Canis lupus) thuộc họ Canidae. Sói xám được coi là loài săn mồi đỉnh cao, nghĩa là chúng đứng đầu chuỗi thức ăn. Sói xám chủ yếu thích sống theo bầy đàn. Kích thước nhóm trung bình là 5-12 con. Điều đặc biệt thú vị là chúng có hệ thống phân cấp thống trị xã hội rất nghiêm ngặt như thể hiện trong Hình 1:



Hình 1. Hệ thống phân cấp của loài sói xám (sự thống trị giảm dần từ trên xuống dưới)

Một bầy sói bao gồm bốn loại:

Những con đầu đàn là một con đực và một con cái, được gọi là alpha. Con alpha chủ yếu chịu trách nhiệm đưa ra quyết định về việc săn bắn, nơi ngủ, thời gian thức dậy, v.v. Các quyết định của alpha được đưa ra cho cả đàn. Tuy nhiên, một số loại hành vi dân chủ cũng đã được quan sát thấy, trong đó một con alpha đi theo những con sói khác trong đàn. Trong các cuộc tụ họp, toàn bộ đàn thừa nhận alpha bằng cách hạ đuôi xuống. Sói alpha cũng được gọi là sói thống trị vì lệnh của nó phải được cả đàn tuân theo [46]. Những con sói alpha chỉ được phép giao phối trong đàn. Điều thú vị là, alpha không nhất thiết phải là thành viên mạnh nhất trong đàn nhưng là người giỏi nhất về mặt quản lý đàn. Điều này cho thấy rằng tổ chức và kỷ luật của một đàn quan trọng hơn nhiều so với sức mạnh của nó.

Cấp độ thứ hai trong hệ thống phân cấp của sói xám là beta. Các beta là những con sói cấp dưới giúp alpha trong việc ra quyết định hoặc các hoạt động khác của bầy. Sói beta có thể là sói đực hoặc sói cái, và anh ta/cô ta có lẽ là ứng cử viên tốt nhất để trở thành alpha trong trường hợp một trong những con sói alpha qua đời hoặc trở nên rất già. Sói beta phải tôn trọng alpha, nhưng cũng phải ra lệnh cho những con sói cấp thấp hơn. Nó đóng vai trò là cố vấn cho alpha và là người kỷ luật cho bầy. Beta củng cố các mệnh lệnh của alpha trong toàn bầy và đưa ra phản hồi cho alpha.

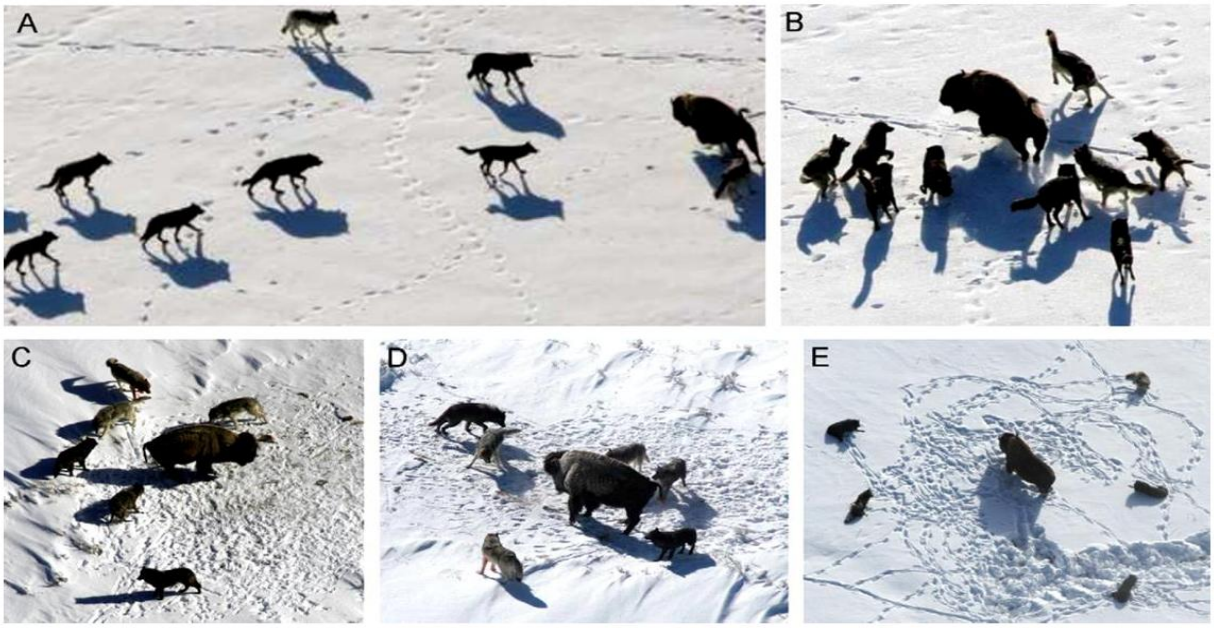
Sói xám xếp hạng thấp nhất là omega. Omega đóng vai trò là vật tế thần. Sói Omega luôn phải khuất phục trước tất cả những con sói thống trị khác. Chúng là những con sói cuối cùng được phép ăn. Có vẻ như omega không phải là một cá thể quan trọng trong bầy, nhưng người ta đã quan sát thấy rằng toàn bộ bầy phải đối mặt với đấu đá nội bộ và các vấn đề trong trường hợp mất omega. Điều này là do sự trút giận dữ và sự thất vọng của tất cả các con sói bởi omega(s). Điều này giúp thỏa mãn toàn bộ bầy và duy trì cấu trúc thống trị. Trong một số trường hợp, omega cũng là người trông trẻ trong bầy.

Nếu một con sói không phải là alpha, beta hoặc omega, nó được gọi là cấp dưới (hoặc delta trong một số tài liệu tham khảo). Sói Delta phải phục tùng alpha và beta, nhưng chúng thống trị omega. Các sói trinh sát, sói canh gác, sói già, sói săn và sói hỗ trợ thuộc nhóm này. Sói trinh sát chịu trách nhiệm canh giữ ranh giới lãnh thổ và cảnh báo bầy khi có nguy hiểm. Sói canh gác bảo vệ và đảm bảo sự an toàn của bầy. Sói già là những con sói có kinh nghiệm, từng là sói alpha hoặc beta. Sói săn săn hỗ trợ alpha và beta trong việc săn mồi và cung cấp thức ăn cho bầy. Cuối cùng, sói hỗ trợ chịu trách nhiệm chăm sóc những con sói yếu, ốm hoặc bị thương trong bầy.

Ngoài hệ thống phân cấp xã hội của loài sói, săn mồi theo nhóm là một hành vi xã hội thú vị khác của loài sói xám. Theo Muro và các cộng sự. Các giai đoạn chính của việc săn sói xám như sau:

- Theo dõi, đuổi bắt và tiếp cận con mồi
- Truy đuổi, quấy rối và bao vây
- Tấn công về phía con mồi

Các bước này được thể hiện ở Hình 2.



Hình 2. Hành vi săn mồi của sói xám: (A) đuổi theo, tiếp cận và theo dõi con mồi (B-D) truy đuổi, quấy rối và bao vây (E) tình hình đứng yên và tấn công

Sau đây, kỹ thuật săn mồi và hệ thống phân cấp xã hội của bầy sói xám được mô hình hóa toán học nhằm thiết kế thuật toán GWO và thực hiện tối ưu hóa.

2.2. Mô hình toán học và thuật toán

Trong phần này, các mô hình toán học của hệ thống phân cấp xã hội, theo dõi, bao vây và tấn công con mồi được cung cấp. Sau đó, thuật toán GWO được phác thảo.

2.2.1. Hệ thống phân cấp xã hội

Để mô hình hóa toán học hệ thống phân cấp xã hội của loài sói khi thiết kế GWO, chúng tôi coi giải pháp phù hợp nhất là alpha (α). Do đó, giải pháp tốt thứ hai và thứ ba được đặt tên tương ứng là beta (β) và delta (δ). Các giải pháp ứng viên còn lại được coi là omega (ω). Trong thuật toán GWO, việc săn mồi (tối ưu hóa) được hướng dẫn bởi α, β và δ. Những con sói (ω) đi theo ba con sói này.

2.2.2. Mô phỏng bao vây con mồi

Như đã đề cập, sói xám bao vây con mồi trong khi săn mồi. Để mô hình hóa hành vi bao vây về mặt toán học, các phương trình sau được đề xuất:

$$\text{Độ lệch: } \vec{D} = | \vec{C} \cdot \overrightarrow{Xp}(t) - \vec{X}(t) | \quad (2.1)$$

$$\text{Vị trí sói tại vòng lặp tiếp theo: } \vec{X}(t+1) = \overrightarrow{Xp}(t) - \vec{A} \cdot \vec{D} \quad (2.2)$$

Trong đó:

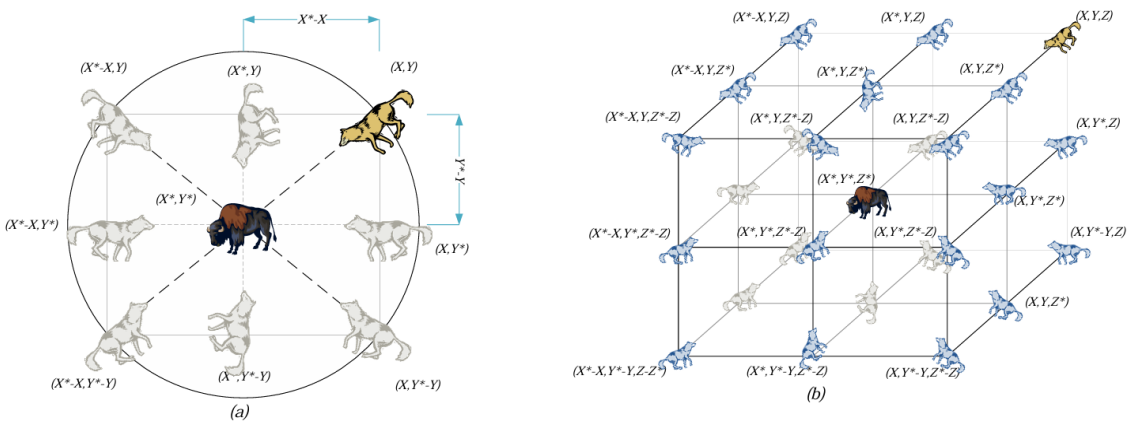
- t chỉ ra vòng lặp hiện tại
- $\overrightarrow{Xp}$ là vector vị trí của con mồi
- $\vec{X}$ chỉ ra vector vị trí của một con sói xám
- $\vec{A}$ và $\vec{C}$ là các vector hệ số được tính như sau:

$$\vec{A} = 2\vec{a} \cdot \overrightarrow{r_1} - \vec{a} \quad (2.3)$$

$$\vec{C} = 2 \cdot \overrightarrow{r_2} \quad (2.4)$$

  - $\vec{a}$: Giá trị giảm tuyến tính từ 2 đến 0 qua các vòng lặp
  - $\overrightarrow{r_1}$,$\overrightarrow{r_2}$: Các giá trị ngẫu nhiên trong khoảng [0, 1].

Để thấy được ảnh hưởng của các phương trình (2.1) và (2.2), một vector vị trí hai chiều và một số láng giềng có thể có được minh họa trong Hình 3 (a). Như có thể thấy trong hình này, một con sói xám ở vị trí (X,Y) có thể cập nhật vị trí của nó theo vị trí của con mồi (X*,Y*). Các vị trí khác nhau quanh mục tiêu tốt nhất có thể đạt được tùy thuộc vào vị trí hiện tại bằng cách điều chỉnh giá trị của các vector $\vec{A}$ và $\vec{C}$. Ví dụ, (X*-X, Y*) có thể đạt được bằng cách thiết lập $\vec{A}$ và $\vec{C}$. Các vị trí cập nhật có thể của một con sói xám trong không gian 3D được minh họa trong Hình 3 (b). Lưu ý rằng các vector ngẫu nhiên $\vec{A}$ và $\vec{C}$ cho phép sói đạt được bất kỳ vị trí nào giữa các điểm được minh họa trong Hình 3. Vì vậy, một con sói xám có thể cập nhật vị trí của nó trong không gian xung quanh con mồi ở bất kỳ vị trí ngẫu nhiên nào bằng cách sử dụng các phương trình (2.1) và (2.2).



Hình 3. Các vectơ vị trí 2D và 3D và các vị trí tiếp theo có thể có của chúng

Khái niệm tương tự có thể được mở rộng ra không gian tìm kiếm với *n* chiều, và các con sói xám sẽ di chuyển trong các hyper-cubes (hoặc hyper-spheres) xung quanh giải pháp tốt nhất đạt được cho đến nay.

2.2.3. Mô phỏng săn mồi

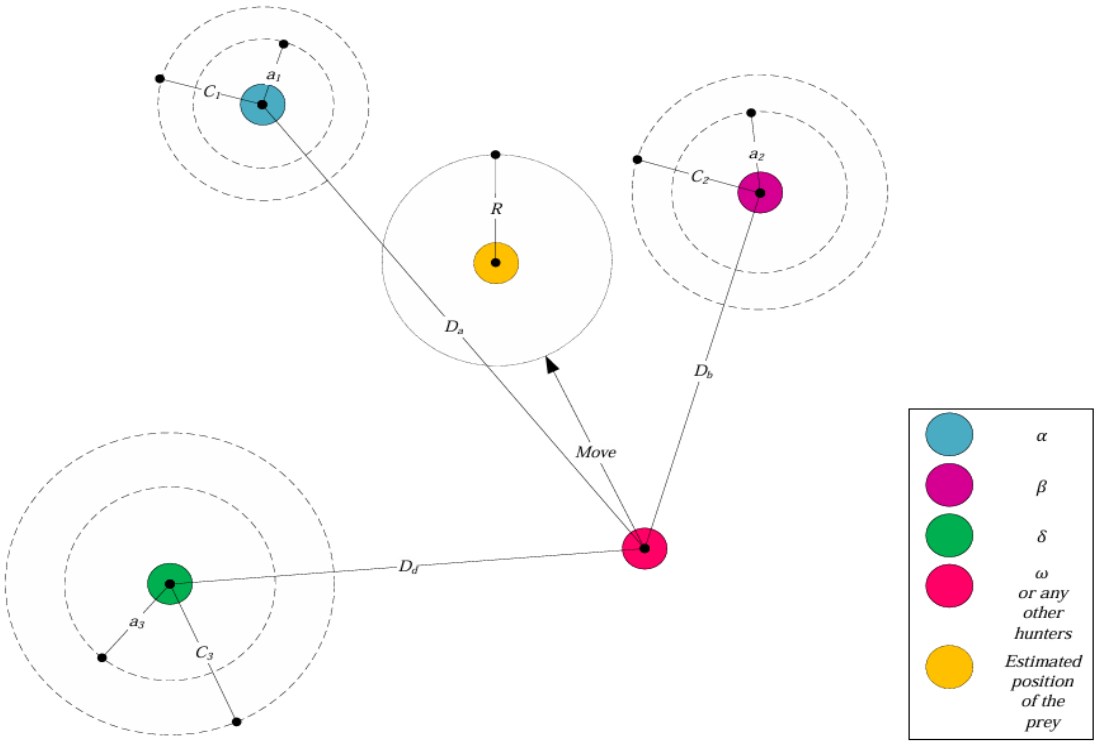
Sói xám có khả năng nhận biết vị trí của con mồi và bao vây chúng. Cuộc săn thường được dẫn dắt bởi alpha. Beta và delta cũng có thể thỉnh thoảng tham gia săn mồi. Tuy nhiên, trong một không gian tìm kiếm trừu tượng, chúng ta không biết gì về vị trí của con mồi tối ưu. Để mô phỏng hành vi săn mồi của sói xám về mặt toán học, chúng tôi cho rằng alpha (giải pháp ứng viên tốt nhất) beta và delta có kiến thức tốt hơn về vị trí tiềm năng của con mồi. Do đó, chúng ta lưu ba giải pháp tốt nhất đầu tiên thu được cho đến nay và yêu cầu các tác nhân tìm kiếm khác (bao gồm cả omega) cập nhật vị trí của chúng theo vị trí của tác nhân tìm kiếm tốt nhất. Các công thức sau đây được đề xuất về vấn đề này.

$$\vec{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \tag{2.5}$$

Trong đó:

- $\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}, \quad \overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}, \quad \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta}$ (2.6)
- $\overrightarrow{D\alpha} = \left| \overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X} \right|, \quad \overrightarrow{D_\beta} = \left| \overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X} \right|, \quad \overrightarrow{D_\delta} = \left| \overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X} \right|$ (2.7)

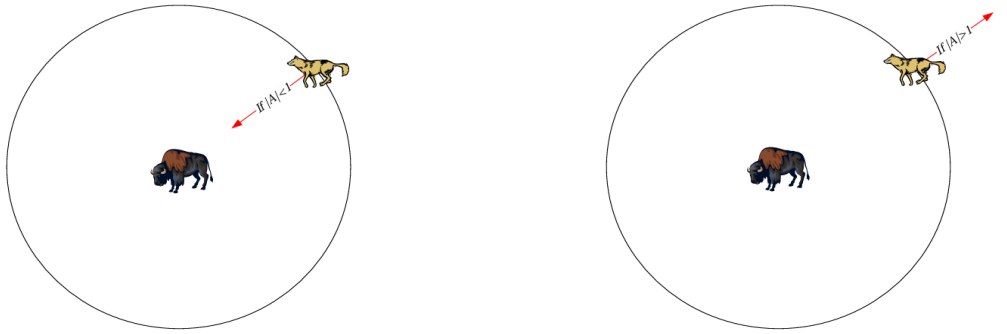
Hình 4 minh họa cách mà một tác nhân tìm kiếm cập nhật vị trí của mình theo alpha, beta và delta trong không gian tìm kiếm 2D. Có thể nhận thấy rằng vị trí cuối cùng sẽ nằm ở một vị trí ngẫu nhiên trong một hình tròn, được xác định bởi các vị trí của alpha, beta và delta trong không gian tìm kiếm. Nói cách khác, alpha, beta và delta ước lượng vị trí của con mồi, và các con sói khác cập nhật vị trí của chúng ngẫu nhiên quanh con mồi.



Hình 4. Cập nhật vị trí trong GWO

2.2.4. Tấn công con mồi (Khai thác)

Như đã đề cập ở trên, các con sói xám hoàn thành việc săn mồi bằng cách tấn công con mồi khi nó ngừng di chuyển. Để mô hình hóa toán học việc tiếp cận con mồi, chúng ta giảm giá trị của $\vec{a}$. Lưu ý rằng phạm vi dao động của $\vec{A}$ cũng bị giảm bởi $\vec{a}$. Nói cách khác, $\vec{A}$ là một giá trị ngẫu nhiên trong khoảng [-a, a], trong đó a giảm dần từ 2 xuống 0 qua các vòng lặp. Khi các giá trị ngẫu nhiên của $\vec{A}$ nằm trong khoảng [-1, 1], vị trí tiếp theo của một tác nhân tìm kiếm có thể nằm ở bất kỳ vị trí nào giữa vị trí hiện tại của nó và vị trí của con mồi. Hình 5 (a) cho thấy |A|<1 buộc các con sói phải tấn công về phía con mồi.



(a) *(b)*

Hình 5. Tấn công con mồi so với tìm kiếm con mồi

Với các phép toán đã được đề xuất cho đến nay, thuật toán GWO cho phép các tác nhân tìm kiếm cập nhật vị trí của chúng dựa trên vị trí của alpha, beta, và delta; và tấn công về phía con mồi. Tuy nhiên, thuật toán GWO có xu hướng bị đình trệ ở các giải pháp cục bộ với các phép toán này. Mặc dù cơ chế bao vây đã được đề xuất có thể thể hiện khả năng khám phá ở một mức độ nào đó, nhưng GWO cần thêm các phép toán để nhấn mạnh việc khám phá.

2.2.5. Tìm kiếm con mồi (Khai phá)

Các con sói xám chủ yếu tìm kiếm dựa trên vị trí của alpha, beta và delta. Chúng phân tán khỏi nhau để tìm kiếm con mồi và hội tụ để tấn công con mồi. Để mô hình hóa toán học sự phân tán, chúng ta sử dụng $\vec{A}$ với các giá trị ngẫu nhiên lớn hơn 1 hoặc nhỏ hơn -1 để buộc các tác nhân tìm kiếm phân tán khỏi con mồi. Điều này nhấn mạnh việc khám phá và cho phép thuật toán GWO tìm kiếm toàn cục. Hình 5(b) cũng cho thấy |A|>1 buộc các con sói xám phải phân tán khỏi con mồi, với hy vọng tìm được con mồi khỏe hơn. Một thành phần khác của GWO giúp thúc đẩy khám phá là $\vec{C}$. Như có thể thấy trong phương trình (2.4), vector $\vec{C}$ chứa các giá trị ngẫu nhiên trong khoảng [0, 2]. Thành phần này cung cấp trọng số ngẫu nhiên cho con mồi để ngẫu nhiên nhấn mạnh (C > 1) hoặc giảm nhẹ (C < 1) ảnh hưởng của con mồi trong việc xác định khoảng cách trong Phương trình (2.1). Điều này giúp GWO thể hiện hành vi ngẫu nhiên hơn trong suốt quá trình tối ưu hóa, thúc đẩy khám phá và tránh các cực trị cục bộ. Cũng cần lưu ý rằng C không giảm tuyến tính như A. Chúng tôi cố ý yêu cầu C cung cấp các giá trị ngẫu nhiên liên tục trong suốt quá trình để nhấn mạnh việc khám phá không chỉ trong các vòng lặp ban đầu mà còn trong các vòng lặp cuối. Thành phần này rất hữu ích trong trường hợp đình trệ ở cực trị cục bộ, đặc biệt là trong các vòng lặp cuối.

Vector $\vec{C}$ cũng có thể được coi là ảnh hưởng của các vật cản trong việc tiếp cận con mồi trong tự nhiên. Nói chung, các vật cản trong tự nhiên xuất hiện trong các con đường săn mồi của sói và thực tế ngăn chúng tiếp cận con mồi nhanh chóng và dễ dàng. Đây chính xác là điều mà

vector $\vec{C}$ thực hiện. Tùy vào vị trí của một con sói, nó có thể ngẫu nhiên đưa ra trọng số cho con mồi và làm cho việc tiếp cận con mồi trở nên khó khăn và xa hơn đối với các con sói, hoặc ngược lại.

Tóm lại, quá trình tìm kiếm bắt đầu bằng cách tạo ra một quần thể ngẫu nhiên các con sói xám (các giải pháp ứng viên) trong thuật toán GWO. Qua các vòng lặp, các con sói alpha, beta và delta ước lượng vị trí có thể có của con mồi. Mỗi giải pháp ứng viên cập nhật khoảng cách của nó với con mồi. Tham số a giảm từ 2 xuống 0 để nhấn mạnh việc khám phá và khai thác, tương ứng. Các giải pháp ứng viên có xu hướng phân tán khỏi con mồi khi |A|>1 và hội tụ về phía con mồi khi |A|<1. Cuối cùng, thuật toán GWO được kết thúc khi đạt được một tiêu chí kết thúc.

Giải mã của thuật toán GWO:

Khởi tạo quần thể sói xám X_i(i = 1,2,…,n)

Khởi tạo a, A và C

Tính toán mức độ phù hợp của từng tác nhân tìm kiếm

X_α= trình tìm kiếm tốt nhất

X_β= trình tìm kiếm tốt thứ 2

X_δ= trình tìm kiếm tốt thứ 3

while(t < số lần lặp lại tối đa)

    for mỗi tác nhân tìm kiếm

        Cập nhật vị trí của tác nhân tìm kiếm hiện tại theo phương trình (2.5)

    end for

    Cập nhật a, A và C

    Tính toán độ thích nghi của tất cả các tác nhân tìm kiếm

    Cập nhật X_α, X_β and X_δ

    t=t+1

end while

return X_α

Để thấy cách GWO có thể giải quyết các bài toán tối ưu lý thuyết, có thể lưu ý một số điểm sau:

- Hệ thống phân cấp xã hội được đề xuất giúp GWO lưu giữ các giải pháp tốt nhất đã tìm được trong suốt quá trình lặp.
- Cơ chế bao vây được đề xuất xác định một khu vực hình tròn xung quanh các giải pháp, có thể mở rộng ra các không gian chiều cao hơn dưới dạng một siêu cầu.
- Các tham số ngẫu nhiên A và C giúp các giải pháp ứng viên có các siêu cầu với bán kính ngẫu nhiên khác nhau.
- Phương pháp săn mồi được đề xuất cho phép các giải pháp ứng viên xác định vị trí có thể có của con mồi.
- Khám phá và khai thác được đảm bảo nhờ các giá trị thích nghi của tham số a và A.
- Các giá trị thích nghi của các tham số a và A cho phép GWO chuyển tiếp mượt mà giữa khám phá và khai thác.
- Với giá trị A giảm dần, một nửa số vòng lặp được dành cho khám phá ($|A| \geq 1$) và nửa còn lại dành cho khai thác ($|A| < 1$).
- GWO chỉ có hai tham số chính cần điều chỉnh (a và C).

Có thể tích hợp đột biến và các phép toán tiến hóa khác để mô phỏng toàn bộ chu trình sống của sói xám. Tuy nhiên, chúng tôi đã giữ thuật toán GWO càng đơn giản càng tốt với ít phép toán cần điều chỉnh. Các cơ chế như vậy được khuyến nghị cho công việc nghiên cứu trong tương lai.

3. Thực nghiệm

3.1 Vấn đề của GWO

Trong các lần lặp sau của GWO, sói xám dần tiến gần α sói, làm giảm đa dạng tìm kiếm và dễ hội tụ sớm. Công trình này sử dụng toán tử đột biến Cauchy-Gaussian để tăng đa dạng và cải thiện tìm kiếm cục bộ. Sau mỗi lần lặp, α, β, δ sói được đột biến và so sánh với vị trí ban đầu theo cơ chế tham lam. Cá thể có thể lực tốt hơn được chọn cho lần lặp tiếp theo.

3.2 Đột biến Cauchy-Gaussian

$$U_{leader}(t+1) = X_{leader}\left[1 + \lambda_1 cauchy\left(0, \sigma^2\right) + \lambda_2 Gauss\left(0, \sigma^2\right)\right]$$

$$\sigma = \exp\left(\frac{f(X_{leader}) - f(X_\alpha)}{|f(X_\alpha)|}\right)$$

$$X(t+1) = \begin{cases} U_{leader}(t+1) f(U_{leader}(t+1)) \leq f(X_{leader}) \\ X_{leader} otherwise \end{cases}$$

$$\lambda_1 = 1 - \frac{t^2}{T^2} \qquad \lambda_2 = \frac{t^2}{T^2}$$

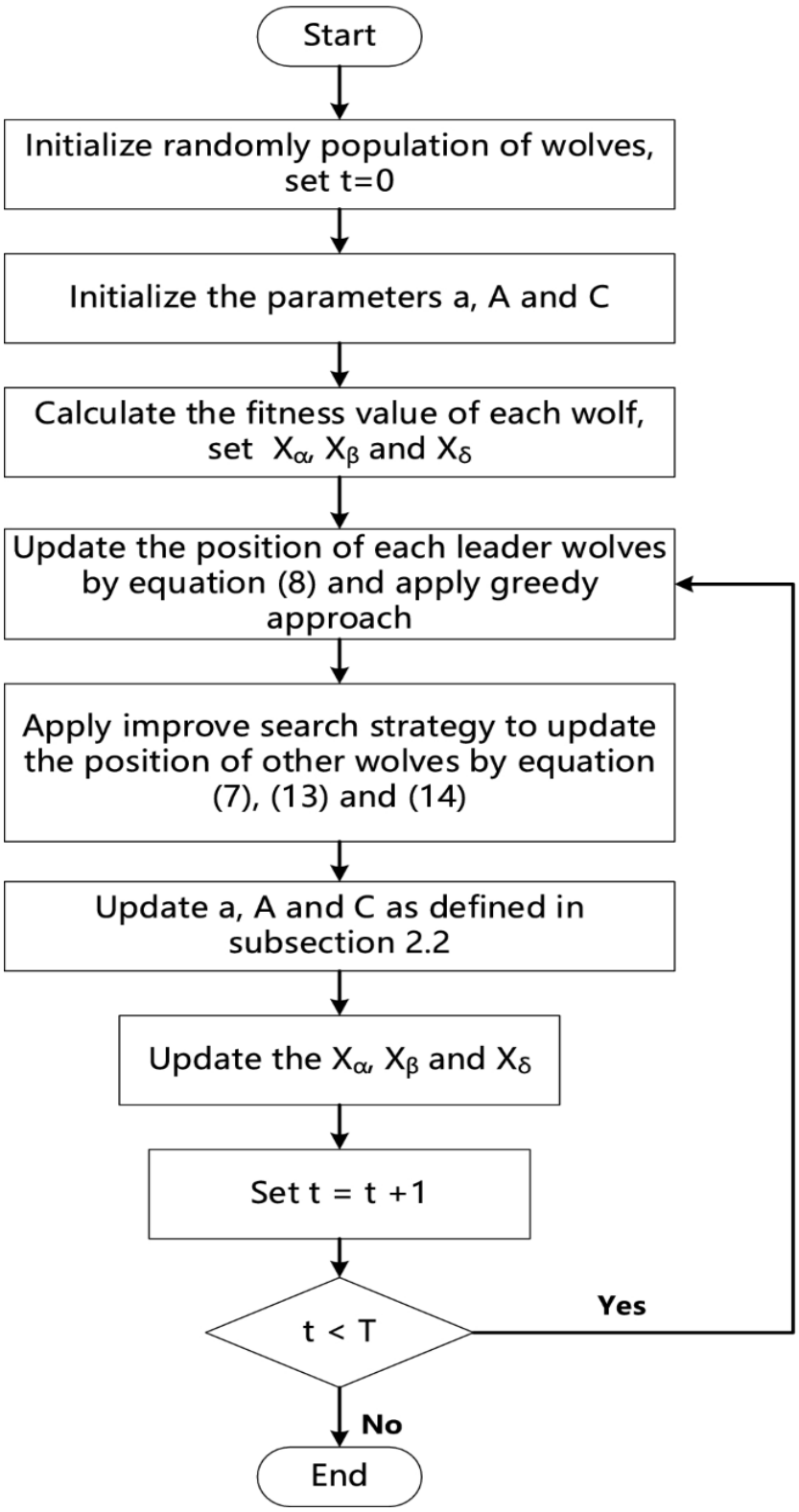
$$U(t+1) = \begin{cases} X_{rand}(t) - r_1 \times |X_{rand}(t) - 2 \times r_2 \times X(t)|, r_5 \geq 0.5 \\ X_\alpha(t) - X_{avg}(t) - r_3 \times (lb + r_4 \times (ub - lb)), r_5 < 0.5 \end{cases}$$

$$X(t+1) = \begin{cases} U(t+1), f(U(t+1) \leq f(X(t))) \\ X(t), otherwise \end{cases}$$

3.3 Thuật toán GG – GWO

Initialize *the population of grey wolves randomly within the search space*
Initialize *parameters a , A and C*
Initialize *t=1,the iteration number*
Calculate *the fitness of each grey wolf*
Select x_α = *fittest wolf of the pack*
x_β = *second best wolf*
x_δ = *third best wolf*
While $t <$ *maximum number of iterations*
for *each leader wolf*
update the position by equation (8) and apply greedy approach
end
for *other wolves*
apply improve search strategy to update the position by equation (7) , (13) and (14)
end
update *a , A and C as defined in subsection 2.2*
update *the* x_α , x_β *and* x_δ
$t = t + 1$
end

Hình 1 Mã giả của thuật toán



Hình 2 Sơ đồ của thuật toán

3.4 Mô phỏng và kết quả

Thiết kế thử nghiệm

Thí nghiệm sử dụng 16 hàm điểm chuẩn, gồm 5 hàm đơn phương thức (F1–F5), 6 hàm đa phương thức (F6–F11), và 5 hàm đa phương thức kích thước cố định (F12–F16). Hàm đơn phương thức kiểm tra khả năng hội tụ và khám phá, hàm đa phương thức đánh giá tìm kiếm toàn cầu và tối ưu hóa cục bộ, còn hàm đa phương thức kích thước cố định kiểm tra độ ổn định thuật toán.

Function formula	Dim	Range	f_{min}
$F_1(x) = \sum_{i=1}^{n} x_i^2$	30	[– 100, 100]	0
$F_2(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	[– 100, 100]	0
$F_3(x) = \max\{\|x_i\|, 1 \le i \le n\}$	30	[– 100, 100]	0
$F_4(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	30	[– 100, 100]	0
$F_5(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$	30	[– 1.28, 1.28]	0

Bảng 1

Function formula	Dim	Range	f_{min}
$F_6(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\|x_i\|}\right)$	30	[– 500, 500]	– 418.9829 × 5
$F_7(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[– 5.12, 5.12]	0
$F_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	[– 32, 32]	0
$F_9(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[– 30, 30]	0
$F_{10}(x) = 0.1\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2(1 + \sin^2(2\pi x_n))\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	30	[– 50, 50]	0
$F_{11}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	[– 50, 50]	0

Bảng 2

Function formula	Dim	Range	f_{min}
$F_{12}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	2	[− 65, 65]	1
$F_{13}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	4	[− 5, 5]	0.0003
$F_{14}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	[− 5, 5]	− 1.0316
$F_{15}(x) = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$	2	[− 5, 5]	0.398
$F_{16}(x) = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2 \right) \right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2 \right) \right]$	2	[− 2, 2]	3

Bảng 3

Công trình này tiến hành các thí nghiệm so sánh gồm hai phần để kiểm tra hiệu quả của cách tiếp cận: so sánh với các thuật toán cổ điển (GWO, PSO, WOA, SSA, FFA) và so sánh với các thuật toán cải tiến (GLF-GWO, IGWO, mGWO). Thí nghiệm được thực hiện trên PC (Intel i7-4790, RAM 8GB, ổ cứng 903GB) với Python 3.6.8. Tất cả các thuật toán chạy độc lập 30 lần trên mỗi hàm, quy mô dân số 30 và 200 lần lặp tối đa. Kết quả thu được gồm giá trị Tốt nhất, Trung bình, Tệ nhất và Độ lệch chuẩn (SD).

Kết quả thí nghiệm và phân tích: Độ chính xác hội tụ

Hiệu quả tối ưu hóa được kiểm tra bằng cách so sánh thuật toán đề xuất với GWO, PSO,WOA, SSA, FFA, IGWO, mGWO và GLF-GWO trên 16 hàm điểm chuẩn.

Function	Algorithm	Best	Ave
F1	PSO	2.42E−03	2.03E−02
	WOA	6.06E−43	1.29E−38
	SSA	1.18E−01	4.12E+00
	FFA	3.42E−23	4.62E−14
	GWO	7.78E−17	1.35E−15
	CG-GWO	**1.49E−162**	**2.81E−150**
F2	PSO	3.63E+01	1.31E+02
	WOA	1.88E+04	4.13E+04
	SSA	7.63E+02	1.89E+03
	FFA	3.52E+03	4.25E+03
	GWO	3.48E−04	3.05E−02
	CG-GWO	**1.37E−142**	**3.56E−133**
F3	PSO	8.94E−01	1.32E+00
	WOA	6.09E−04	3.20E+01
	SSA	5.83E+00	1.13E+01
	FFA	4.34E−02	6.15E−01
	GWO	1.24E−04	9.01E−04
	CG-GWO	**1.01E−77**	**1.04E−72**
F4	PSO	3.35E−03	1.82E−02
	WOA	2.98E−02	1.36E−01

Function	Algorithm	Best	Ave
F5	SSA	1.82E−01	3.94E+00
	FFA	3.15E−02	5.16E−01
	GWO	1.32E−04	2.45E−01
	CG-GWO	**7.42E−05**	**1.51E−05**
	PSO	6.68E−02	8.95E−01
	WOA	5.91E−05	1.83E−03
	SSA	9.15E−02	1.82E−01
	FFA	3.48E−02	4.15E−02
	GWO	6.27E−04	2.13E−03
	CG-GWO	**4.76E−05**	**2.51E-04**
F6	PSO	− 5.38E+03	− 4.40E + 03
	WOA	**− 1.26E+04**	− 1.09E + 04
	SSA	− 8.64E+03	− 7.31E+03
	FFA	− 7.62+03	− 6.82E+03
	GWO	− 8.07E+03	− 6.30E+03
	CG-GWO	**− 1.26E+04**	− **1.26E+04**
F7	PSO	5.92E+01	1.07E+02
	WOA	**0**	1.89E−15

Function	Algorithm	Best	Ave
	SSA	1.82E+01	3.90E+01
	FFA	1.23E+01	2.61E+01
	GWO	2.25E+00	1.33E+01
	CG-GWO	**0**	**0**
	PSO	3.44E−02	3.64E−01
	WOA	**4.44E−16**	6.96E−15
F8	SSA	2.05E+00	3.53E+00
	FFA	3.51E−01	4.34E−01
	GWO	3.28E−09	9.36E−09
	CG-GWO	**4.44E−16**	**4.44E−16**
	PSO	2.33E−02	9.22E−01
	WOA	**0**	1.59E−02
F9	SSA	3.38E−01	8.84E−01
	FFA	3.18E−01	5.68E−01
	GWO	2.66E−15	3.31E−03
	CG-GWO	**0**	**0**
F10	PSO	1.15E−03	1.31E−02
	WOA	6.09E−02	1.94E−01

Function	Algorithm	Best	Ave
F11	SSA	4.97E+00	5.13E+01
	FFA	4.53E−01	6.15E−01
	GWO	1.94E−04	2.36E−01
	CG-GWO	**6.43E−05**	**1.16E−04**
	PSO	1.35E−01	3.58E−01
	WOA	6.59E−04	7.12E−04
	SSA	3.21E−03	5.15E−03
	FFA	6.52E−04	7.36E−04
	GWO	2.48E−02	3.56E−03
	CGGWO	**6.25E−08**	**3.36E−07**
F12	PSO	**9.98E−01**	1.36E+00
	WOA	**9.98E−01**	1.89E+00
	SSA	**9.98E−01**	1.59E+00
	FFA	**9.98E−01**	1.68E+00
	GWO	**9.98E−01**	2.70E+00
	CG-GWO	**9.98E−01**	**9.98E−01**
F13	PSO	3.31E−04	2.75E−03
	WOA	3.13E−04	7.03E−04

Function	Algorithm	Best	Ave
F14	SSA	4.88E−04	1.69E−03
	FFA	3.84E−04	3.45E−03
	GWO	3.08E−04	1.72E−03
	CG-GWO	**3.07E−04**	**3.74E−04**
	PSO	**− 1.03E+00**	**− 1.03E+00**
	WOA	**− 1.03E+00**	**− 1.03E+00**
	SSA	**− 1.03E+00**	**− 1.03E+00**
	FFA	**− 1.03E+00**	**− 1.03E+00**
	GWO	**− 1.03E+00**	**− 1.03E+00**
	CG-GWO	**− 1.03E+00**	**− 1.03E+00**
F15	PSO	**3.98E−01**	**3.98E−01**
	WOA	**3.98E−01**	**3.98E−01**
	SSA	**3.98E−01**	**3.98E−01**
	FFA	**3.98E−01**	**3.98E−01**
	GWO	**3.98E−01**	**3.98E−01**
	CG-GWO	**3.98E−01**	**3.98E−01**
F16	PSO	**3.00E+00**	**3.00E+00**
	WOA	**3.00E+00**	**3.00E+00**

Function	Algorithm	Best	Ave
	SSA	**3.00E+00**	**3.00E+00**
	FFA	**3.00E+00**	**3.00E+00**
	GWO	**3.00E+00**	**3.00E+00**
	CG-GWO	**3.00E+00**	**3.00E+00**

Function	Algorithm	Best	Ave
	GWO	7.78E−17	1.35E−15
	mGWO	1.03E−21	1.20E−20
F1	IGWO	4.81E−25	9.61E−24
	GLF-GWO	3.81E−33	2.85E−31
	CG-GWO	**1.49E−162**	**2.81E−150**
	GWO	3.48E−04	3.05E−02
	mGWO	1.95E−05	2.96E−03
F2	IGWO	9.61E−07	9.13E−04
	GLF-GWO	2.34E−04	7.66E−03
	CG-GWO	**1.37E−142**	**3.56E−133**
	GWO	1.24E−04	9.01E−04
	mGWO	6.90E−06	3.61E−05
F3	IGWO	5.39E−06	3.73E−05
	GLF-GWO	3.95E−05	1.44E−04
	CG-GWO	**1.01E−77**	**1.04E−72**
	GWO	1.32E−04	2.45E−01
	mGWO	4.39E−04	2.96E−01
F4	IGWO	3.24E−01	8.71E−01
	GLF-GWO	1.10E−04	7.77E−04
	CG-GWO	**7.42E−05**	**1.51E−05**

Function	Algorithm	Best	Ave
	GWO	6.27E−04	2.13E−03
	mGWO	4.09E−04	1.67E−03
F5	IGWO	7.22E−04	1.74E−03
	GLF-GWO	9.02E−04	2.90E−03
	CG-GWO	**4.76E−05**	**2.51E−04**
	GWO	− 8.07E+03	− 6.30E+03
	mGWO	− 8.00E+03	− 5.92E+03
F6	IGWO	− 4.45E+03	− 3.80E+03
	GLF-GWO	− 1.00E+04	− 8.81E+03
	CG-GWO	− **1.26E+04**	− **1.26E+04**
	GWO	2.25E+00	1.33E+01
	mGWO	4.55E−13	8.52E+00
F7	IGWO	5.41E+00	4.18E+01
	GLF-GWO	5.68E−14	2.82E+00
	CG-GWO	**0**	**0**
	GWO	3.28E−09	9.36E−09
F8	mGWO	5.64E−12	2.20E−11
	IGWO	1.57E−13	6.29E−13

Function	Algorithm	Best	Ave
	GLF-GWO	3.60E−14	4.51E−14
	CG-GWO	**4.44E−16**	**4.44E−16**
	GWO	2.66E−15	3.31E−03
	mGWO	**0**	1.98E−03
F9	IGWO	**0**	6.76E−03
	GLF-GWO	**0**	4.77E−03
	CG-GWO	**0**	**0**
	GWO	1.94E−04	2.36E−01
	mGWO	6.68E−04	2.04E−01
F10	IGWO	3.31E−01	8.43E−01
	GLF-GWO	6.57E−05	4.08E−04
	CG-GWO	**6.43E−05**	**1.16E−04**
	GWO	2.48E−02	3.56E−03
	mGWO	1.25E−06	5.15E−06
F11	IGWO	3.12E−06	6.15E−06
	GLF-GWO	6.52E−07	7.19E−07
	CG-GWO	**6.25E−08**	**3.36E−07**
F12	GWO	**9.98E−01**	2.70E +00

Function	Algorithm	Best	Ave
	mGWO	**9.98E−01**	2.25E +00
	IGWO	**9.98E−-01**	4.00E +00
	GLF-GWO	**9.98E−01**	**9.98E−01**
	CG-GWO	**9.98E−01**	**9.98E−01**
	GWO	3.08E−04	1.72E−03
	mGWO	3.08E−04	3.11E−03
F13	IGWO	3.08E−04	1.09E−03
	GLF-GWO	**3.07E−04**	1.80E−03
	CG-GWO	**3.07E−04**	**3.74E−04**
	GWO	− **1.03E+00**	− **1.03E +00**
	mGWO	− **1.03E+00**	− **1.03E +00**
F14	IGWO	− **1.03E+00**	− **1.03E +00**
	GLF-GWO	**− 1.03E+00**	− **1.03E +00**
	CG-GWO	**− 1.03E+00**	− **1.03E +00**
	GWO	**3.98E−01**	**3.98E−01**
	mGWO	**3.98E−01**	**3.98E−01**
F15	IGWO	**3.98E−01**	**3.98E−01**
	GLF-GWO	**3.98E−01**	**3.98E−01**

Function	Algorithm	Best	Ave
F16	CG-GWO	**3.98E−01**	**3.98E−01**
	GWO	**3.00E+00**	**3.00E+00**
	mGWO	**3.00E+00**	**3.00E+00**
	IGWO	**3.00E+00**	**3.00E+00**
	GLF-GWO	**3.00E+00**	**3.00E+00**
	CG-GWO	**3.00E+00**	**3.00E+00**

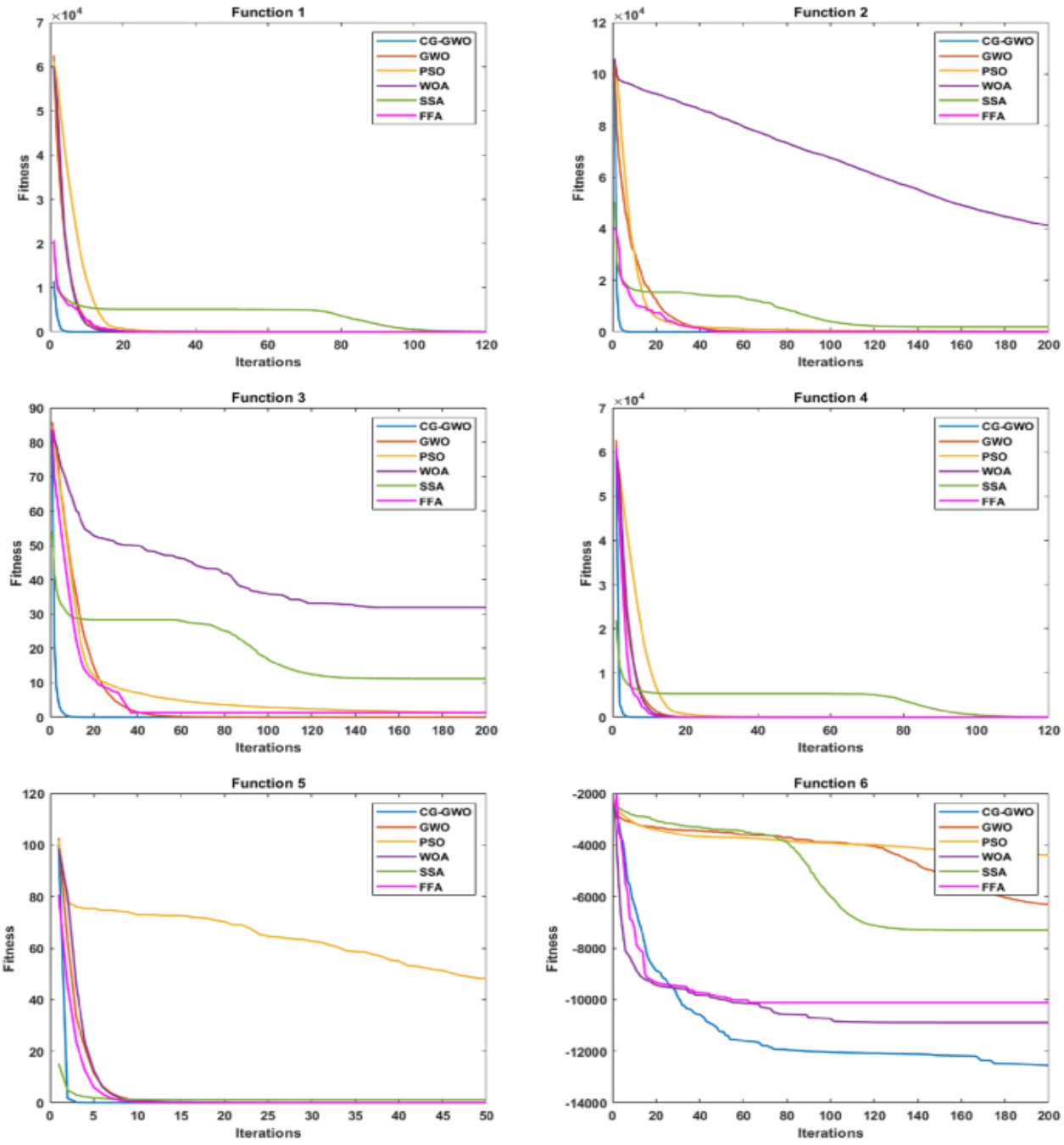
Bảng đầu tiên cho thấy thuật toán CG-GWO đạt giá trị tối ưu, trung bình và tồi tệ nhất ở mức tốt nhất trên tất cả các hàm điểm chuẩn. Trên F12, cả thuật toán CG-GWO và bốn thuật toán cổ điển khác đều đạt giá trị tối ưu, nhưng CG-GWO vượt trội về giá trị trung bình, tồi tệ nhất và độ lệch chuẩn. Trên các hàm F1, F2, F3, F4, F7, F9 và F10, CG-GWO thể hiện sự vượt trội tuyệt đối so với các thuật toán cổ điển, với các chỉ số cao hơn đáng kể. Với F5 và F13, dù không quá nổi bật, CG-GWO vẫn vượt các thuật toán cổ điển. Trên F6 và F8, cả CG-GWO và WOA đạt giá trị tối ưu, nhưng CG-GWO ổn định hơn.

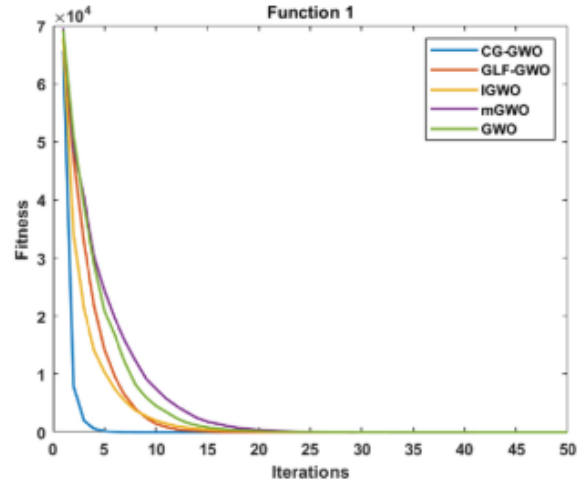
Bảng sau cho thấy CG-GWO vượt trội rõ rệt trên các hàm đơn phương thức so với các thuật toán GWO cải tiến. Trên F1, F2 và F3, các chỉ số của CG-GWO đều vượt xa. Trên F4 và F5, CG-GWO thể hiện sự ổn định vượt trội. Với các hàm đa phương thức như F6, F7, F8 và F10, CG-GWO đạt giá trị tối ưu và các số liệu thống kê đều tốt hơn các thuật toán khác. Trên F9, mặc dù nhiều thuật toán tìm được giá trị tối ưu, CG-GWO vẫn chứng tỏ vượt trội.

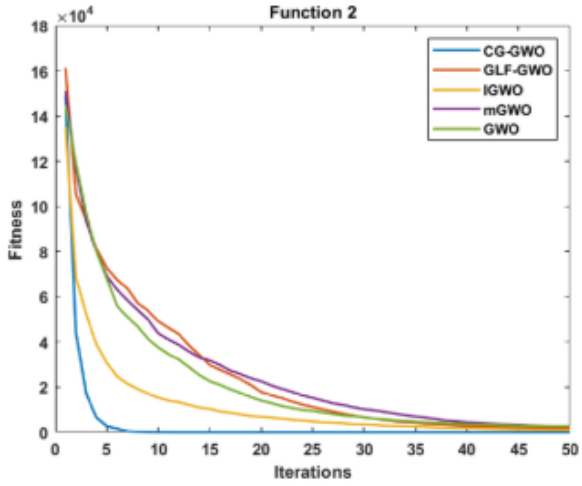
Với các hàm đa phương thức kích thước cố định, CG-GWO đạt độ chính xác tối ưu ngang ngửa các thuật toán khác. Trên F12, tất cả các thuật toán đều tìm được giá trị tối ưu, nhưng CG-GWO ổn định hơn với độ lệch chuẩn nhỏ. Trên F13, dù cả GLF-GWO và CG-GWO đều đạt giá trị tối ưu, CG-GWO cho thấy sự tập trung và ổn định vượt trội.

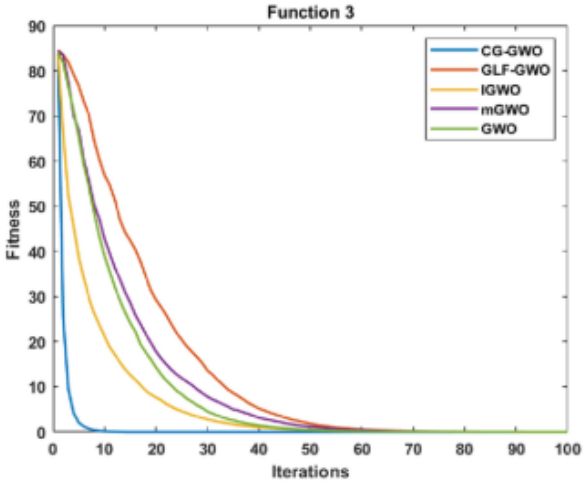
Phân tích tốc độ hội tụ

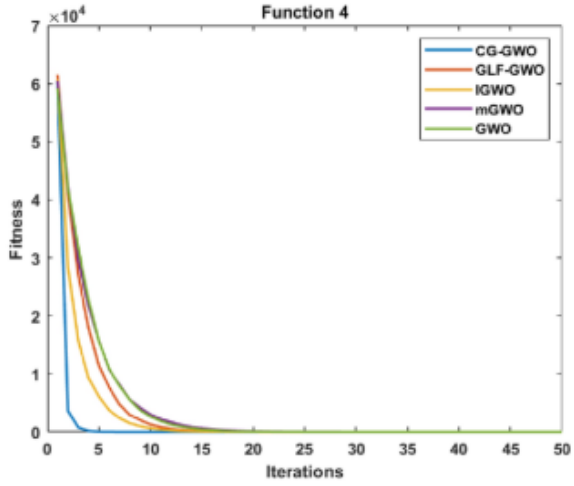
Thí nghiệm tốc độ hội tụ được thực hiện nhằm quan sát trực quan hiệu quả và tốc độ hội tụ của thuật toán CG-GWO so với các thuật toán cổ điển và các thuật toán GWO cải tiến. Các đường cong hội tụ trên 16 hàm điểm chuẩn, với trục hoành là số lần lặp và trục tung là giá trị phù hợp, được thể hiện trong 2 hình sau.

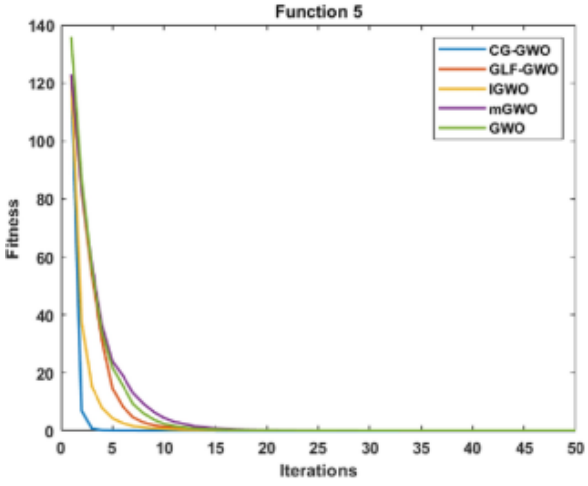


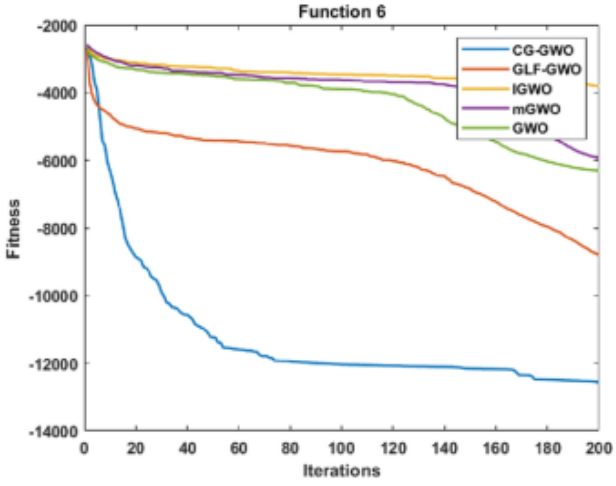

Function 1
×10^4
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO


Function 2
×10^4
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO


Function 3
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO


Function 4
×10^4
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO


Function 5
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO


Function 6
Fitness
Iterations
CG-GWO
GLF-GWO
IGWO
mGWO
GWO

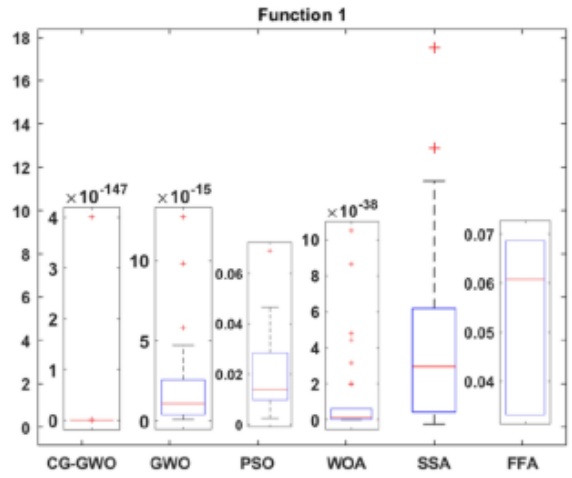
Nó có thể được thấy từ **hình đầu tiên** rằng đường cong hội tụ của thuật toán CG-GWO thấp hơn các thuật toán tối ưu hóa cổ điển khác trong thí nghiệm. Độ chính xác và tốc độ hội tụ trên 16 hàm điểm chuẩn đã được cải thiện đáng kể. Đối với các hàm đơn phương thức, như trong F1 và F4, CG-GWO hội tụ nhanh đến giá trị tối ưu, trong khi SSA không đạt được giá trị tối ưu sau 100 lần lặp. Cách tiếp cận của chúng tôi đạt giá trị tối ưu trong 10 lần lặp, còn GWO mất đến 50 lần lặp. PSO, WOA, SSA và FFA không đạt giá trị tối ưu sau 200 lần lặp. Đối với các hàm đa phương thức, GWO và PSO không có xu hướng hội tụ, SSA hội tụ vào mức tối ưu cục bộ, trong khi WOA và CG-GWO có độ dốc lớn, với CG-GWO hội tụ nhanh hơn WOA sau 40 lần lặp. Các hàm F7–F8 cho thấy PSO, SSA và FFA không hội tụ sau 200 lần lặp, trong khi CG-GWO hội tụ nhanh hơn. Đối với các hàm F12-F16, tất cả các thuật toán có thể hội tụ gần giá trị tối ưu, nhưng CG-GWO hội tụ nhanh hơn và WOA hội tụ chậm nhất.

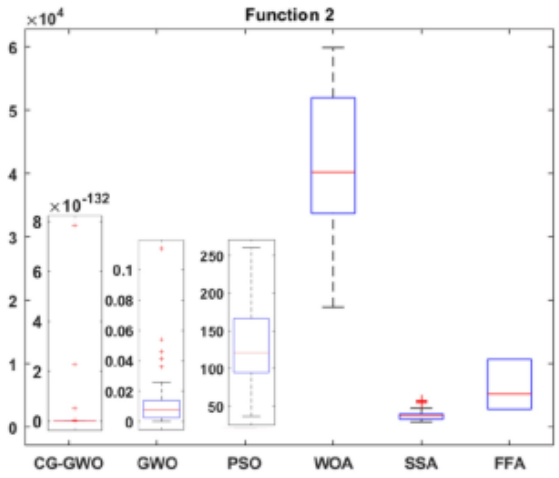
Nó có thể được thấy từ **hình sau** rằng thuật toán CG-GWO vượt trội so với GWO truyền thống và các thuật toán cải tiến. Trên các hàm đơn phương thức F1–F5, tất cả các thuật toán hội tụ đến giá trị tối ưu, nhưng CG-GWO hội tụ nhanh hơn. Trên các hàm đa phương thức F6 và F7, CG-GWO khám phá gần giá trị tối ưu lý thuyết sau 200 lần lặp và tốc độ hội tụ nhanh hơn các thuật toán cải tiến khác. Trên các hàm F8, F9, F10, F11, CG-GWO hội tụ nhanh hơn. Đối với các hàm F12 và F16, tốc độ hội tụ của tất cả các thuật toán tương đương, nhưng độ chính xác của CG-GWO cao hơn. Trên các hàm F13–F15, CG-GWO hội tụ nhanh hơn các thuật toán khác.

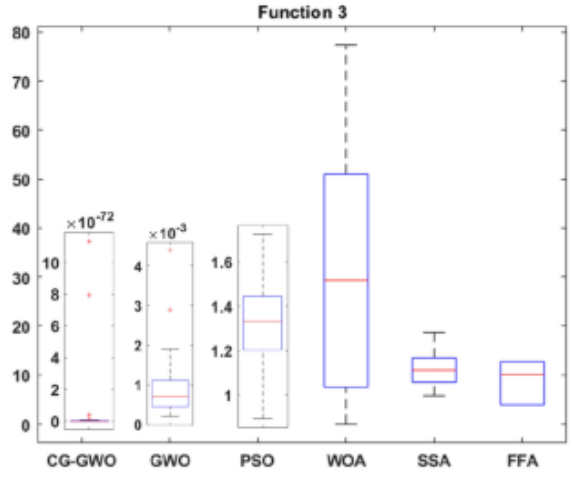
Vì vậy, CG-GWO vượt trội về độ chính xác và tốc độ hội tụ, đồng thời cải thiện khả năng tìm kiếm toàn cầu, tránh rơi vào tối ưu cục bộ. Các thí nghiệm chứng minh hiệu quả của các cải tiến và tính ưu việt của thuật toán trong giải quyết các vấn đề tối ưu hóa phức tạp.

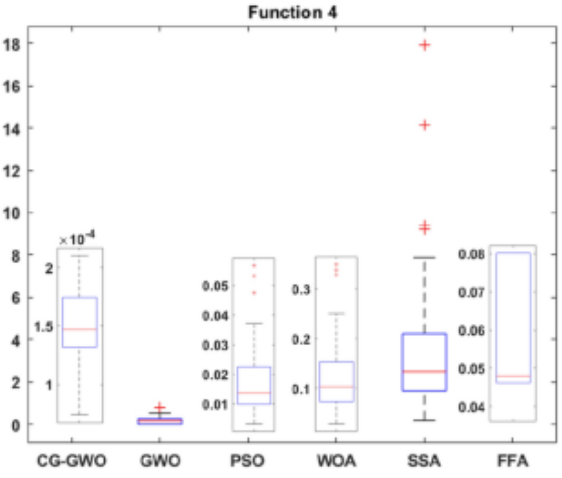
Phân tích hiệu suất thuật toán

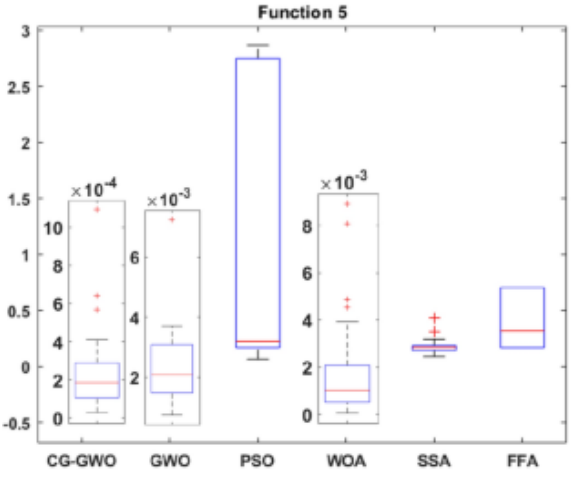
Để đánh giá khả năng tối ưu hóa và độ ổn định của thuật toán cải tiến, công trình này vẽ biểu đồ hộp của tất cả các thuật toán trên 16 hàm điểm chuẩn. Phân tích so sánh được thực hiện dựa trên bốn điểm chính: giá trị trung bình, bốn điểm thấp hơn và ngoại lệ trong biểu đồ hộp. Công trình tiến hành 30 thí nghiệm độc lập trên tất cả các thuật toán. Vì kết quả thực nghiệm của các thuật toán khác nhau khá khác biệt, nên các hệ tọa độ khác nhau được sử dụng để so sánh khi vẽ biểu đồ hộp, nhằm quan sát kết quả so sánh một cách trực quan hơn. Kết quả thí nghiệm được trình bày trong **2 hình sau.**

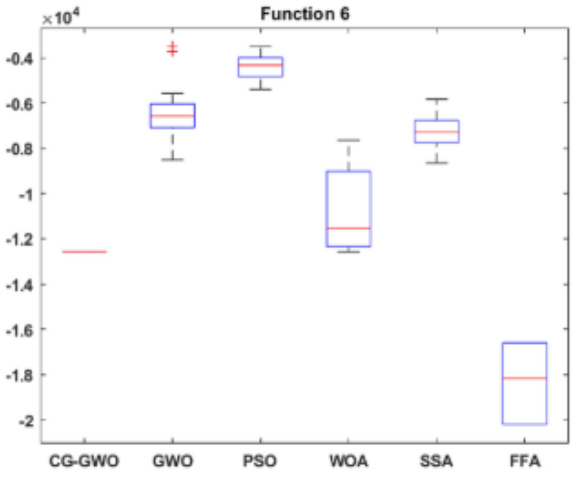

Function 1
CG-GWO
GWO
PSO
WOA
SSA
FFA

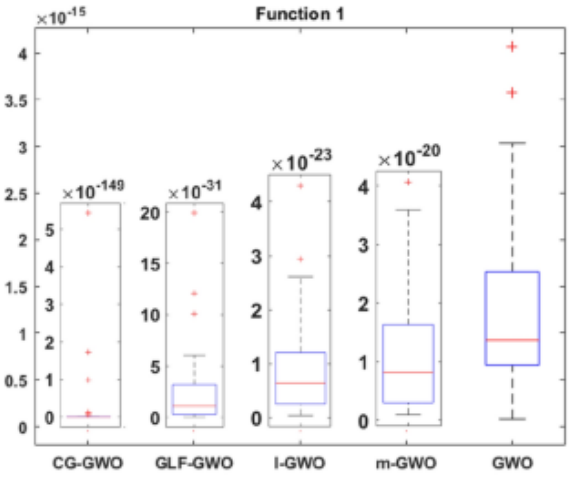

Function 2
CG-GWO
GWO
PSO
WOA
SSA
FFA

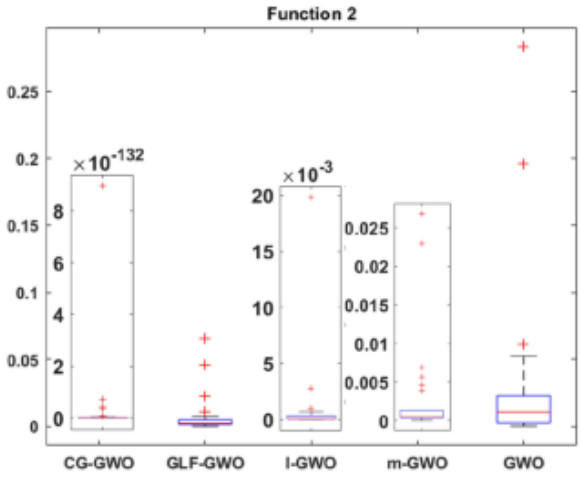

Function 3
CG-GWO
GWO
PSO
WOA
SSA
FFA

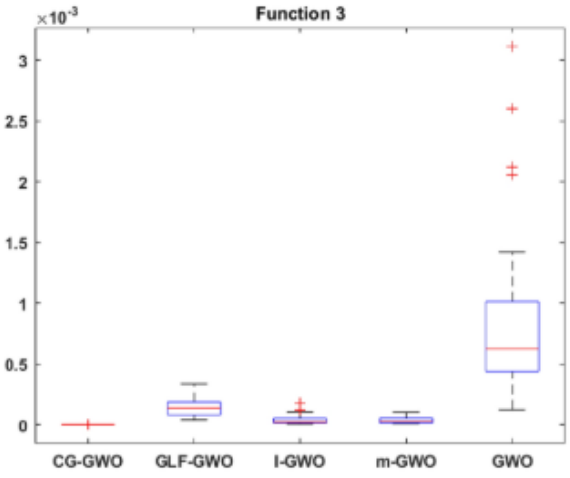

Function 4
CG-GWO
GWO
PSO
WOA
SSA
FFA

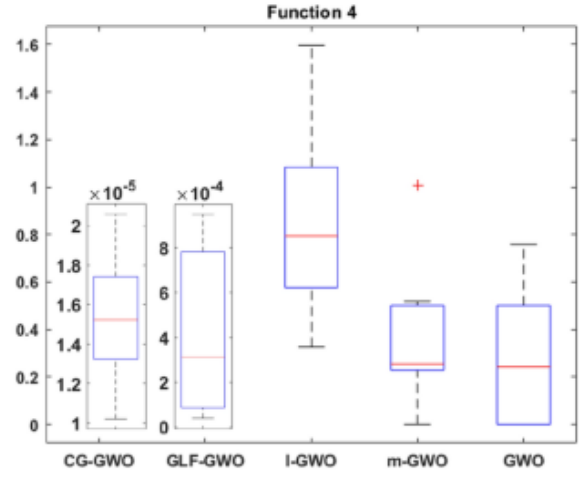

Function 5
CG-GWO
GWO
PSO
WOA
SSA
FFA

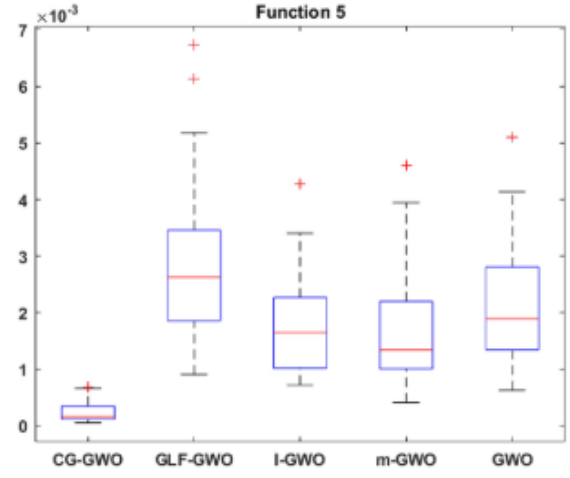

Function 6
CG-GWO
GWO
PSO
WOA
SSA
FFA

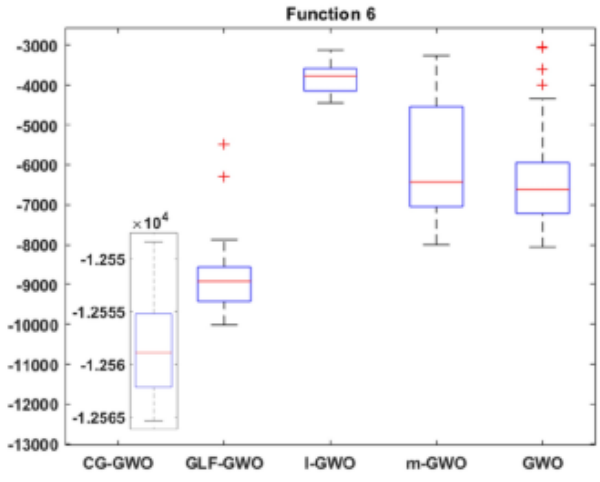

Function 1
×10^-15
×10^-149
×10^-31
×10^-23
×10^-20
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO


Function 2
×10^-132
×10^-3
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO


Function 3
×10^-3
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO


Function 4
×10^-5
×10^-4
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO


Function 5
×10^-3
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO


Function 6
×10^4
CG-GWO
GLF-GWO
I-GWO
m-GWO
GWO

Nó có thể được nhìn thấy từ **hình đầu tiên** rằng thuật toán CG-GWO có độ phân tán yếu hơn trong quá trình tối ưu hóa và giá trị tối ưu hóa của nó tập trung hơn. Trên các hàm đơn phương thức F1–F5, thứ tự độ lớn của kết quả tối ưu hóa của cách tiếp cận của chúng tôi nhỏ hơn nhiều so với các thuật toán tối ưu hóa cổ điển khác trong thí nghiệm. Kết quả thí nghiệm phản ánh sự vượt trội về độ chính xác tối ưu hóa và độ ổn định cao hơn của thuật toán cải tiến. Trên các hàm đa phương thức F6–F11, cách tiếp cận của chúng tôi tập trung hơn và có ít ngoại lệ hơn so với các thuật toán tối ưu hóa cổ điển khác trong thử nghiệm. Kết quả thí nghiệm đã xác minh tính mạnh mẽ của cách tiếp cận của chúng tôi về khả năng tìm kiếm toàn cầu. Trên các hàm đa phương thức kích thước cố định F12–F13, thuật toán CG-GWO và thuật toán GWO có tác dụng tối ưu hóa tương tự. Chúng tốt hơn nhiều so với các thuật toán tối ưu hóa cổ điển khác trong thử nghiệm. Tuy nhiên, cách tiếp cận của chúng tôi có ít ngoại lệ hơn, các giá trị tối ưu hóa tập trung hơn và kết quả tối ưu hóa tốt hơn. Như thể hiện trong F14–F16, CG-GWO có ít ngoại lệ hơn và ổn định hơn.

Như trong **hình sau**, hầu hết các thuật toán GWO cải tiến đều cho thấy sự vượt trội so với thuật toán GWO truyền thống. Trên các hàm đơn phương thức F1–F5, thuật toán CG-GWO có hiệu quả tối ưu hóa tốt hơn. Trên các hàm điểm chuẩn F3 và F5, mức độ tương phản của mỗi thuật toán tương đối gần, nhưng cách tiếp cận của chúng tôi cho thấy độ ổn định hội tụ cao hơn và có ít ngoại lệ hơn. Trên các hàm đa phương thức F6–F11, chỉ có tính ưu việt của thuật toán GLF-GWO là gần với cách tiếp cận của chúng tôi. Mặc dù khả năng tìm kiếm toàn cầu của thuật toán GLF-GWO đã được nâng cao do cơ chế tìm kiếm chuyến bay Levy, nhưng nó tạo ra nhiều ngoại lệ hơn. Trên các hàm đa phương thức kích thước cố định F12–F16, thuật toán GLF-GWO tương đối gần với cách tiếp cận của chúng tôi. Thuật toán GLF-GWO tốt hơn thuật toán CG-GWO về phạm vi thay đổi giá trị tối ưu, nhưng cách tiếp cận của chúng tôi có ít ngoại lệ hơn nhiều. Kết quả thử nghiệm thể hiện khả năng tìm kiếm toàn cầu tốt hơn và độ ổn định cao hơn của cách tiếp cận .

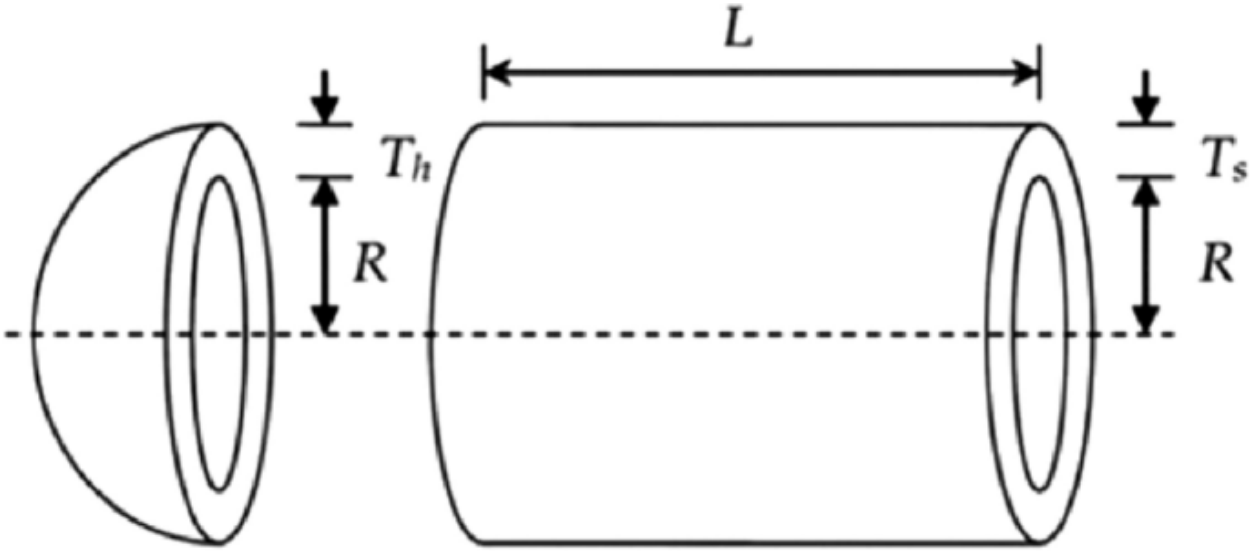
Nghiên cứu điển hình về ứng dụng thực tế

Nghiên cứu điển hình về ứng dụng thực tế này liên quan đến việc đánh giá hiệu suất của chín thuật toán tối ưu hóa khác nhau, bao gồm PSO, WOA, SSA, FFA, GWO, I-GWO, m-GWO, GLF-GWO và CG-GWO, trong một ứng dụng thực tế kỹ thuật: thiết kế bình áp lực.

Bình áp lực, được sử dụng trong nhiều ngành công nghiệp, thường có hai dạng hình học chính là hình cầu và hình trụ. Các mạch hình trụ này có thể được định hướng theo chiều dọc hoặc ngang. Bình áp lực dạng trụ có nhiều ứng dụng quan trọng, bao gồm tháp phân đoạn, tháp công tắc tơ, lò phản ứng và máy tách dọc.

Mục tiêu của bài toán là giảm chi phí vật liệu tổng thể khi thiết kế bình áp lực có hình dạng trụ, được gia cố bằng các đầu hình bán cầu ở cả hai đầu, như mô tả trong Hình 10. Công thức toán học liên quan đến thiết kế bình áp lực này sẽ bao gồm các yếu tố như kích thước, tải trọng, vật liệu và các yếu tố liên quan đến hình học của bình áp lực.

Việc sử dụng các thuật toán tối ưu hóa như PSO, WOA, SSA, FFA, GWO, I-GWO, m-GWO, GLF-GWO và CG-GWO sẽ giúp tìm ra các tham số tối ưu của thiết kế này nhằm đạt được chi phí thấp nhất và hiệu quả cao nhất, đồng thời đảm bảo tính ổn định và an toàn cho bình áp lực trong các ứng dụng thực tế.



4. Kết luận

Bài tìm hiểu này này trình bày một thuật toán tối ưu SI, lấy cảm hứng từ sói xám. Phương pháp được đề xuất mô phỏng hệ thống phân cấp xã hội và hành vi săn mồi của sói xám. Trong các nghiên cứu hai mươi chín hàm kiểm tra đã được sử dụng để so sánh hiệu suất của thuật toán được đề xuất về các khía cạnh như khám phá, khai thác, tránh các cực trị cục bộ và hội tụ. Các kết quả cho thấy GWO có thể cung cấp các kết quả cạnh tranh cao so với các thuật toán heuristics nổi tiếng như PSO, GSA, DE, EP và ES. Đầu tiên, kết quả trên các hàm đơn điệu cho thấy khả năng khai thác vượt trội của thuật toán GWO. Thứ hai, khả năng khám phá của GWO đã được xác nhận qua kết quả trên các hàm đa cực. Thứ ba, kết quả của các hàm kết hợp cho thấy khả năng tránh cực trị cục bộ cao. Cuối cùng, phân tích hội tụ của GWO đã xác nhận sự hội tụ của thuật toán này.

Hơn nữa, kết quả từ các bài toán thiết kế kỹ thuật cũng cho thấy thuật toán GWO có hiệu suất cao trong các không gian tìm kiếm khó khăn và chưa được biết đến. Thuật toán GWO cuối cùng đã được áp dụng vào một bài toán thực tế trong kỹ thuật quang học. Kết quả của bài toán này cho thấy sự cải thiện đáng kể NDBP so với các phương pháp hiện tại, chứng minh tính khả thi của thuật toán được đề xuất trong việc giải quyết các vấn đề thực tế. Cũng cần lưu ý rằng kết quả từ các bài toán bán thực và thực tế cũng chứng minh rằng GWO có thể thể hiện hiệu suất cao không chỉ trong các bài toán không ràng buộc mà còn trong các bài toán có ràng buộc.

Tham khảo

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm intelligence: from natural to artificial systems: OUP USA, 1999.

[2] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," Computational Intelligence Magazine, IEEE, vol. 1, pp. 28 39, 2006.

[3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Neural Networks, 1995. Proceedings., IEEE International Conference on, 1995, pp. 1942-1948.

[4] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," Evolutionary Computation, IEEE Transactions on, vol. 1, pp. 67-82, 1997.

[5] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in Robots and Biological Systems: Towards a New Bionics?, ed: Springer, 1993, pp. 703-712.

[6] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in IEEE swarm intelligence symposium, 2006, pp. 12-14.

[7] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey Wolf Optimizer, Advances in Engineering Software , vol. 69, pp. 46-61, 2014