Journal of educational equipment: Education management, Volume 1, Issue 304 (January 2023)
ISSN 1859 - 0810

Ứng dụng Matlab trong giảng dạy thuật toán tối ưu hóa dựa trên tìm kiếm bầy đàn - PSO

Nguyễn Đức Minh*

**TS. Học viện Công nghệ Bưu chính Viễn thông*

Received: 12/12/2023; Accepted: 16/12/2023; Published: 21/12/2023

Abstract: *This article introduces the use of Matlab to teach optimization algorithms based on swarm search (Particle Swarm Optimization - PSO). The content of the article presents the PSO algorithm as well as the algorithm flowchart and pseudocode. The article also presents the use of the Matlab program to teach the PSO algorithm, introduces the PSO optimization functions in Mtalab as well as the steps to program the PSO algorithm.*

Keywords: *Matlab, Optimization Toolbox, PSO algorithm.*

1. Đặt vấn đề

Matlab là một phần mềm mạnh mẽ thường được sử dụng trong nhiều lĩnh vực, bao gồm cả giảng dạy và nghiên cứu thuật toán. Việc sử dụng phần mềm này đã trở nên rất phổ biến với giới khoa học trong và ngoài nước, các kết quả được tính toán và tạo ra bằng Matlab có độ chính xác cao và được biểu diễn rất trực quan sinh động. Sử dụng thành thạo Matlab trong giảng dạy và học tập mang lại rất nhiều lợi ích cho giáo viên, học viên và các nhà nghiên cứu khoa học. Giải thuật PSO (Particle Swarm Optimization) là một thuật toán tối ưu hóa khá phức tạp, việc giảng dạy giải thuật này sẽ trở nên dễ dàng và hiệu quả hơn rất nhiều khi sử dụng công cụ Matlab. Bài báo trình bày về việc ứng dụng Optimization Toolbox trong phần mềm Matlab vào việc giảng dạy giải thuật tối ưu hóa PSO với các nội dung chính như sau:

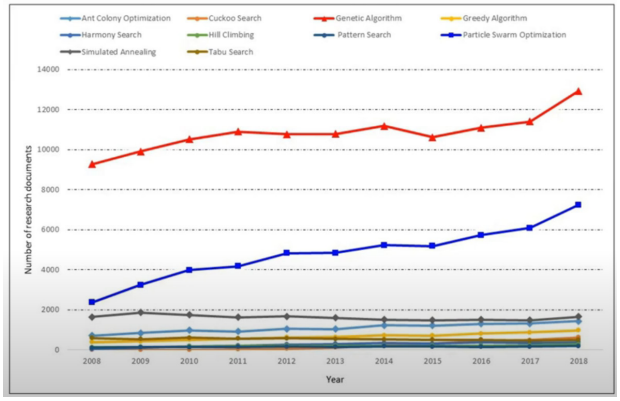
+ Giới thiệu về thuật toán tối ưu hóa PSO.
+ Lưu đồ thuật toán và giả mã của thuật toán PSO
+ Lập trình thuật toán PSO trong Matlab

2. Nội dung nghiên cứu

2.1. Thuật toán PSO

PSO là một giải thuật tối ưu hóa ngẫu nhiên có mức độ phổ biến cao chỉ sau giải thuật di truyền trong việc giải các bài toán tìm kiếm ngẫu nhiên có độ phức tạp lớn theo dữ liệu của Scopus như trong hình vẽ 1. Thuật toán được tác giả Eberhart và Dr.Kennedy [1], mô phỏng theo hành vi của các bầy chim hay các đàn cá trong quá trình di chuyển tìm thức ăn.

Thuật toán PSO tỏ ra có hiệu quả cao trong nhiều bài toán tối ưu hóa đa chiều phức tạp. Thuật toán này được áp dụng rộng rãi để giải các bài toán tối ưu hóa ngưỡng và quy luật hợp nhất trong mạng cảm biến phân tán, tìm đường đi cho rô bốt nhiều chiều trong



Hình 2.1. Dữ liệu của Scopus về số tài liệu nghiên cứu giải thuật thông minh năm 2019

môi trường tĩnh hoặc động, quy hoạch xây dựng hay áp dụng vào hệ thống gợi ý…vv. Đến nay đã có rất nhiều các biến thể của thuật toán PSO bằng việc kết hợp với các giải thuật dựa trên trí tuệ nhân tạo khác. Tuy nhiên trong giải thuật PSO nguyên thủy, mỗi cá thể trong bầy đàn sẽ thay đổi vị trí bằng cách di chuyển đến nhiều vị trí khác nhau trong không gian tìm kiếm cho đến khi tìm được vị trí tốt nhất. Để rõ hơn hãy xem xét một ví dụ: giả sử có một bầy chim đang tìm kiếm thức ăn trong một vùng nào đó và tất cả các con chim đều không biết thức ăn ở đâu. Tuy nhiên, chúng biết là thức ăn cách xa bao nhiêu sau mỗi lần bay đi bay lại và trao đổi thông tin (đây là quá trình lặp). Vậy cách tốt nhất để tìm được thức ăn là gì? câu trả lời đơn giản đó là: bay theo sau những con chim ở gần chỗ thức ăn nhất. PSO phỏng theo kịch bản này và sử dụng nó để giải các bài toán tối ưu. Trong PSO, mỗi một cá thể (particle) có một giá trị thích nghi (fitness value), được đánh giá bằng hàm đo độ thích nghi (fitness function) và một vận tốc

(velocity) để định hướng việc bay để tìm kiếm thức ăn của nó. Các cá thể này sẽ duyệt không gian lời giải của bài toán bài toán bằng cách theo sau các cá thể khác có điều kiện tốt nhất hiện thời.

Thuật toán PSO đầu tiên sẽ khởi tạo một nhóm ngẫu nhiên các cá thể, sau đó tìm kiếm lời giải tối ưu bằng việc cập nhật các cá thể (lần lặp). Trong mỗi thế hệ, mỗi cá thể được cập nhật bởi hai giá trị: giá trị thứ nhất, gọi là P_{best} - là nghiệm tốt nhất đạt được cho tới thời điểm hiện tại - hay là giá trị phù hợp của cá thể tốt nhất trong lần tìm kiếm hiện thời. Giá trị thứ hai, gọi là G_{best} - là nghiệm tốt nhất mà cá thể lân cận cá thể này đạt được cho tới thời điểm hiện tại hay chính là giá trị phù hợp nhất của cá thể tốt nhất trong tất cả các cá thể từ trước đến nay. Nói cách khác, mỗi cá thể trong quần thể cập nhật vị trí của nó theo vị trí tốt nhất của nó và của cá thể trong quần thể tính tới thời điểm hiện tại. Quá trình cập nhật các cá thể dựa trên hai công thức sau:

$$v_{i,m}^{(k+1)} = w.v_{i,m}^{(k)} + c_1 * rand() * (P_{best_{i,m}} - x_{i,m}^{(k)}) + c_2 * rand() * (G_{best_m} - x_{i,m}^{(k)}) \quad (1)$$

$$x_{i,m}^{(k+1)} = x_{i,m}^{(k)} + v_{i,m}^{(k+1)} \quad (2)$$

Ở đây: n là số cá thể trong bầy đàn

d: kích thước quần thể

k: số lần lặp lại

$v_{i,m}^{k}$: vận tốc của cá thể thứ *i* tại thế hệ thứ *k*

w: hệ số trọng lượng quán tính

c_1 và c_2: hệ số gia tốc

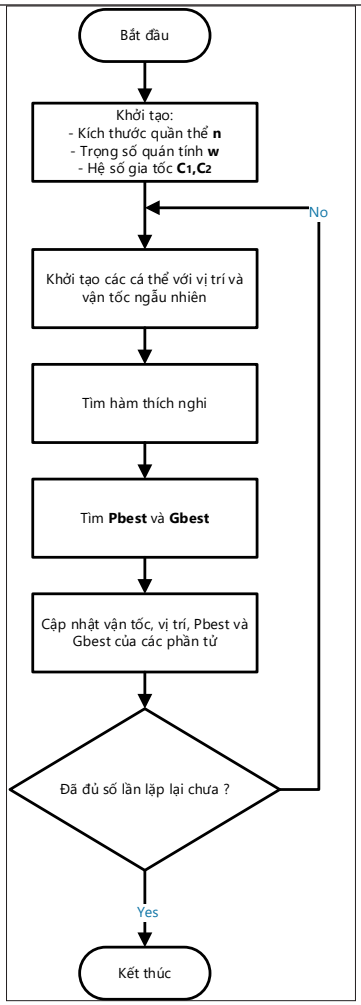
Rand(): là một số ngẫu nhiên trong khoảng [0,1]

$x_{i,m}^{(k)}$: là vị trí cá thể thứ *i* tại thế hệ thứ *k*

P_{best_i}: vị trí tốt nhất của cá thể thứ *i*

G_{best_i}: vị trí tốt nhất của cá thể trong quần thể

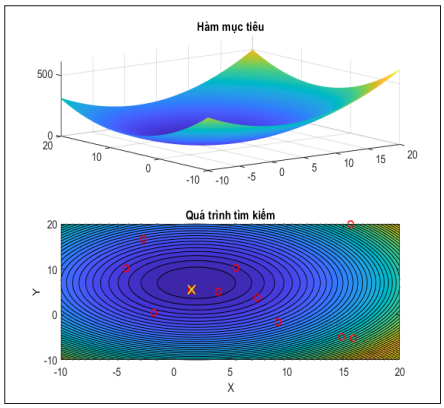
2.2. Lưu đồ thuật toán và giả mã cho giải thuật PSO



Hình 2.2. Lưu đồ thuật toán PSO

Giả mã cho thuật toán PSO được cho như sau:

```
For each Phần_tử
        Khởi tạo Phần_tử
End for
Do
        For each Phần_tử
           Tính Fitness_Value
           If   Fitness_Value < P_best   Then
                P_best = Fitness_Value
           End If
        End For
           If P_best < G_best   Then
                G_best = P_best
           End If
        For each Phần_tử
            Tính Vận_tốc theo công thức  (1)
            Cập nhật vị trí theo công thức (2)
        End  For
While (chưa đạt đến số lần lặp xác định)
```



Hình 2.3.Thuật toán PSO sử dụng Matlab

2.3. Lập trình thuật toán PSO với Matlab

Việc lập trình cho giải thuật PSO trong các ngôn ngữ lập trình máy tính như C hay Python mất rất nhiều thời gian và khá phức tạp dẫn đến khó áp dụng giải thuật này một cách nhanh chóng cho các ứng dụng khoa học khác. Để khắc phục điều này Matlab cung cấp một số hàm cho thuật toán PSO trong Optimization Toolbox như: particleswarm, hay InitialSwarmMatrix,…vv để giải quyết nhanh chóng các bài toán mà không cần phải viết chương trình dài dòng ngoài việc thiết lập các ràng buộc và xây dựng hàm mục tiêu [3]. Chúng ta còn có thể sử dụng một số hàm tích hợp sẵn như rand(), plot(), hoặc scatter() để hiển thị và theo dõi sự di chuyển của các cá thể trong không gian tìm kiếm. Khả năng vẽ đồ thị phong phú của Matlab cũng giúp chúng ta theo dõi và hiển thị trực quan sinh động sự phát triển của quá trình tối ưu. Trong việc giảng dạy, nếu muốn mở rộng, có thể thử nghiệm PSO trên các hàm mục tiêu phức tạp hơn, điều chỉnh các tham số PSO, và so sánh hiệu suất với các thuật toán tối ưu hóa khác. Cũng có thể thêm các chú giải và giải thích vào code để giúp người học hiểu rõ thuật toán và cách nó hoạt động.

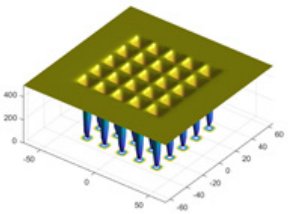
Sau đây là các bước lập trình dùng hàm particleswarm sử dụng thuật toán PSO trong Matlab:

```
- function [best_position, best_value] = PSO(num_particles, num_dimensions, max_iterations)
- % Khởi tạo các particle ngẫu nhiên
particles = rand(num_particles, num_dimensions);
- % Khởi tạo vận tốc ngẫu nhiên
velocities = rand(num_particles, num_dimensions);
- % Khởi tạo vị trí và giá trị tốt nhất
best_position = rand(1, num_dimensions);
best_value = inf;
- % Thực hiện PSO
for iteration = 1:max_iterations
    % Cập nhật vị trí và vận tốc
    % ... (sử dụng các công thức PSO)
- % Đánh giá giá trị tốt nhất
    % ... (sử dụng hàm mục tiêu)
- % Hiển thị quá trình tối ưu
    scatter3(particles(:,1), particles(:,2), particles(:,3), 'filled');
hold on;
plot3(best_position(1), best_position(2), best_position(3), 'rx', 'MarkerSize', 10);
hold off;
drawnow;
end
end
```

Sau đây là một ví dụ về cách sử dụng hàm tối ưu hóa particleswarm trong Matlab khi tối ưu hóa một hàm số De Jong's bậc 5. Đây là một hàm số thường được sử dụng trong việc thử nghiệm các thuật toán tối ưu hóa. Hàm số De Jong's bậc 5 có dạng như sau:

$$f(x) = \sum_{i=1}^{n} x_i^2 \quad (3)$$

Hàm số De Jong thường được sử dụng để kiểm tra hiệu suất của các thuật toán tối ưu hóa và thuật toán tìm kiếm trong không gian đa chiều, vì nó đơn giản và có thể tạo ra các đỉnh và cạnh tốt để kiểm tra tính toàn vẹn và khả năng khám phá của thuật toán. Trong đó x=(x1 ,x2,...,xn) là vector đầu vào có n chiều. Đây là một hàm số đơn giản, chỉ tính tổng bình phương các thành phần của vector đầu vào. Chương trình được cho đơn giản như sau:



Hình 2.4. Hàm số De Jong's bậc 5

```
fun = @dejong5fcn;
nvars = 5;
rng default
lb = [-50;-50];
ub = -lb;
options = optimoptions('particleswarm','SwarmSize',100);
[x,fval,exitflag]=particleswarm(fun,nvars,lb,ub,options)
```

3. Kết luận

Bài báo đã trình bày về thuật toán tối ưu hóa PSO được sử dụng rộng rãi trong nhiều lĩnh vực và việc sử dụng phần mềm Matlab để giảng dạy và mô tả thuật toán. Việc giảng dạy PSO bằng Matlab mang lại nhiều lợi ích, không chỉ là về việc giới thiệu công thức và mã lệnh của chương trình mà còn là việc giúp sinh viên hiểu rõ nguyên lý hoạt động và cách áp dụng thuật toán trong các bài toán thực tế.

Tài liệu tham khảo

1. J.Kennedy and R.Eberhart (1995). "*Particle swarm optimization". In Proceedings of IEEE International Conference on Neural Networks*, pages 1942 -1948. IEEE.

2. Kalyan Veeramachaneni, Lisa Ann Osadciw (2004), *"Dynamic Sensor Management Using Multi Objective Particle Swarm Optimizer"*, *in* Proceedings of SPIE - The International Society for Optical Engineering.