BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

Đề Tài: Thuật toán mô phỏng tự nhiên giải bài toán tối ưu tổ hợp

Giảng viên hướng dẫn: Phạm Hồng Phong

Sinh viên thực hiện: Nguyễn Trung Hiệp

MSV: 69565

Lớp: 65CS3

Hà Nội - 04/2024

Mục Lục

LỜI NÓI ĐẦU

Như chúng ta đã biết, trong khoảng 10 năm trở lại đây, công nghệ thông tin đã bùng nổ và phát triển mạnh mẽ ở nước ta. Có thể nói sự phát triển như vũ bão của khoa học và công nghệ trong thời gian qua đã tạo ra những sản phẩm công nghệ mới và đem lại rất nhiều lợi ích cho cuộc sống. Nó đang chiếm phần lớn trong việc phục vụ của nhiều ngành nghề cũng như phục vụ đời sống của con người. Cùng với đó là loài người chúng ta luôn muốn tìm hiểu, cải tiến và khắc phục các nhược điểm của các bài toán có trong cuộc sống. Ví dụ như bài toán lập lịch, người bán hàng,… cho nên trong bài báo cáo này em tập trung nghiên cứu về 2 thuật toán được áp dụng nhiều để giải quyết các bài toán tối ưu, cùng với đó là so sánh với các thuật toán khác để thấy được sự khác biệt mà 2 thuật toán mang lại.

Trân trọng.

Phần 1: Tổng quan về đề tài

1.1 Giới thiệu

- Như đã nói ở trên thì bài báo cáo này của em sẽ nghiên cứu về 2 thuật toán chủ đạo đó là thuật toán di truyền (Genetic Algorithms - GA), thuật toán tối ưu hóa đàn kiến (Ant Colony Optimization - ACO).
  a. GA: là một thuật toán tối ưu hóa nguồn cảm hứng từ tự nhiên, mô phỏng quá trình tiến hóa sinh học. Thuật toán này sử dụng các cơ chế di truyền như chọn lọc tự nhiên, lai ghép (crossover), và đột biến (mutation) để tạo ra các thế hệ giải pháp mới từ thế hệ hiện tại. GA bắt đầu từ một quần thể các cá thể (các giải pháp) ngẫu nhiên, sau đó lặp đi lặp lại quá trình chọn lọc, lai ghép, và đột biến để cải thiện chất lượng của quần thể. Thuật toán này phù hợp cho các bài toán tối ưu hóa liên tục hoặc tổ hợp, ví dụ như tối ưu hóa thiết kế, tối ưu hóa đa mục tiêu, và vấn đề phân công
  b. ACO: là một thuật toán tối ưu hóa nguồn cảm hứng từ tự nhiên, dựa trên hành vi tìm kiếm thức ăn của đàn kiến. Thuật toán này được Marco Dorigo đề xuất vào đầu những năm 1990. Trong ACO, các kiến khám phá không gian tìm kiếm và xây dựng các giải pháp dựa trên phép đánh giá chất lượng của các giải pháp và một chất pheromone mà chúng tiết ra. Chất pheromone này giúp hướng dẫn các kiến khác đến các giải pháp tốt hơn. Thuật toán này thường được sử dụng để giải quyết các bài toán tối ưu hóa tổ hợp, chẳng hạn như bài toán người bán hàng (Travelling Salesman Problem - TSP), bài toán lập lịch, và các bài toán tuyến đường.

1.2 Lý do chọn đề tài

- Ứng dụng Rộng Rãi: Cả ACO và GA đều có khả năng ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau như tối ưu hóa trong lĩnh vực logistics, lập lịch sản xuất, vận tải, thiết kế mạng, và nhiều bài toán khoa học khác. Việc tìm hiểu sâu về các thuật toán này có thể mở ra nhiều cơ hội áp dụng thực tế.
- Cải Thiện và Đổi Mới: Cả hai thuật toán đều cho phép không gian lớn để đổi mới và cải tiến. Có thể tập trung vào cải thiện hiệu quả tính toán của chúng, hoặc phát triển các biến thể mới phù hợp hơn với các loại bài toán cụ thể.

- Giải quyết Bài toán Phức Tạp: Thuật toán ACO và GA đặc biệt hữu ích trong việc giải quyết các bài toán tối ưu hóa phức tạp, nơi các phương pháp truyền thống không hiệu quả. Chọn các thuật toán này làm đề tài nghiên cứu có thể giúp giải quyết những thách thức thực tế trong các ngành công nghiệp và nghiên cứu.
- Nhu cầu Thực Tiễn và Thương Mại: Cả hai thuật toán có nhiều ứng dụng thực tế có thể chuyển giao sang các sản phẩm và dịch vụ thương mại. Tìm hiểu về chúng có thể cung cấp nền tảng để phát triển các giải pháp công nghệ cao, góp phần vào sự phát triển kinh tế.
- Tính Đa Dạng và Linh Hoạt: ACO và GA cung cấp một khung tư duy linh hoạt cho các nhà khoa học và kỹ sư để giải quyết nhiều loại bài toán khác nhau. Nghiên cứu về chúng giúp phát triển kỹ năng giải quyết vấn đề và sáng tạo trong nhiều hoàn cảnh khác nhau.
- Cộng Đồng Nghiên Cứu Sôi Nổi: Cả hai lĩnh vực này có một cộng đồng nghiên cứu lớn và sôi động, với nhiều hội nghị, workshop, và tạp chí chuyên ngành, cung cấp cơ hội tốt để trao đổi kiến thức và kết nối với các nhà nghiên cứu khác.

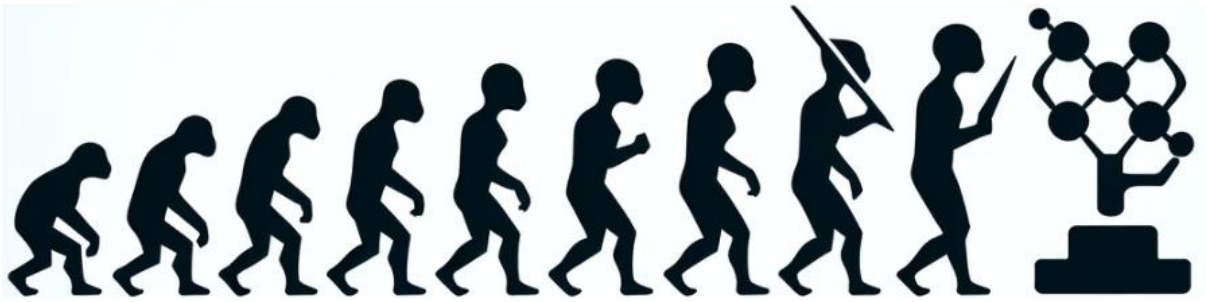
1.3 Phạm vi nghiên cứu

1. Đối Tượng Nghiên Cứu
    - ACO: Tập trung vào giải quyết các bài toán tối ưu hóa tổ hợp, như bài toán người bán hàng (TSP) và bài toán lập lịch.
    - GA: Áp dụng cho các bài toán tối ưu hóa liên tục và đa mục tiêu, chẳng hạn như trong thiết kế kỹ thuật và sinh học tính toán.
2. Cải Tiến Kỹ Thuật
    - Phát triển và thử nghiệm các cải tiến thuật toán nhằm cải thiện hiệu suất, độ chính xác, và thời gian tính toán.
3. Ứng Dụng Thực Tế
    - Phát triển các ứng dụng cụ thể của ACO và GA trong lĩnh vực như robotics và bioinformatics.
4. Phương Pháp Đánh Giá
    - So sánh hiệu quả của ACO và GA với các thuật toán khác trong điều kiện thực nghiệm nhằm đánh giá ưu nhược điểm.
5. Môi Trường Thực Nghiệm
    - Sử dụng các công cụ và môi trường phát triển phổ biến Python để triển khai và kiểm thử thuật toán.

Phần 2: Thuật toán di truyền (GA) và thuật toán tối ưu hóa đàn kiến (ACO)

2.1 Thuật toán di truyền (GA)

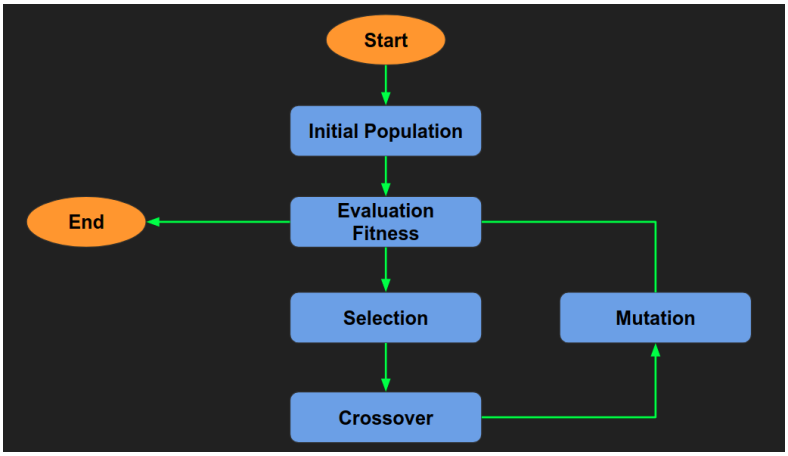
- John Holland, người phát minh ra Thuật toán Di truyền vào những năm 1960 tại Đại học Michigan. Mục tiêu ban đầu của Holland là tạo ra một mô hình máy tính mô phỏng các nguyên tắc của tiến hóa sinh học để nghiên cứu các quá trình thích nghi.
- Nguyên lý cơ bản của Thuật toán Di truyền: Giải thích cách GA mô phỏng quá trình tiến hóa tự nhiên thông qua chọn lọc tự nhiên, lai ghép, và đột biến. Mô tả cách mà GA tạo ra các thế hệ cá thể mới để tìm kiếm các giải pháp tối ưu cho một bài toán cụ thể. Vai trò của di truyền trong việc chuyển giao các đặc tính giữa các thế hệ và vai trò của đột biến trong việc duy trì sự đa dạng di truyền, từ đó giúp thuật toán khám phá ra những iải pháp mới và hiệu quả hơn.



2.1.1 Các thành phần cơ bản của GA

1. Mã hóa (Encoding)
   a. Mục đích: Giải thích tầm quan trọng của việc mã hóa đối với hiệu quả của GA. Mã hóa là quá trình biểu diễn các biến quyết định của bài toán thành một chuỗi, thường được gọi là chromosome.
   b. Các loại mã hóa: Phổ biến nhất là mã hóa nhị phân, nhưng còn có mã hóa số nguyên, số thực, và mã hóa dựa trên đối tượng cho các bài toán phức tạp hơn. Mô tả ưu và nhược điểm của mỗi phương pháp.

2. Khởi tạo quần thể (Population Initialization)
    a. Mô tả quá trình: Quần thể ban đầu của GA thường được sinh ra một cách ngẫu nhiên để đảm bảo sự đa dạng. Quần thể gồm nhiều cá thể, mỗi cá thể đại diện cho một giải pháp tiềm năng.
    b. Tầm quan trọng: Khởi tạo quần thể là bước đầu tiên và quan trọng để khám phá không gian giải pháp. Số lượng cá thể trong quần thể có ảnh hưởng đến khả năng tìm kiếm và hiệu suất của GA.
3. Hàm thích nghi (Fitness Function)
    a. Xác định: Hàm thích nghi là hàm đánh giá mức độ "thích hợp" của mỗi cá thể trong quần thể, thường dựa trên chất lượng của giải pháp mà cá thể đó biểu diễn.
    b. Vai trò: Hàm thích nghi quyết định cá thể nào sẽ "sống sót" và được chọn để sinh sản trong thế hệ tiếp theo. Hàm thích nghi phải được thiết kế cẩn thận để đảm bảo rằng nó phản ánh chính xác mục tiêu của bài toán.
4. Chọn lọc (Selection)
    a. Mục đích: Chọn lọc là quá trình chọn ra cá thể từ quần thể hiện tại để tạo ra cá thể cho quần thể thế hệ tiếp theo.
    b. Phương pháp: Giới thiệu các kỹ thuật chọn lọc phổ biến như Roulette Wheel, Tournament Selection, và Stochastic Universal Sampling. Mỗi phương pháp có ưu và nhược điểm riêng trong cách thức tạo đa dạng di truyền.
5. Lai ghép (Crossover)
    a. Xác định: Lai ghép là quá trình kết hợp đặc điểm của hai cá thể cha mẹ để tạo ra cá thể con.
    b. Cách thực hiện: Mô tả các kiểu lai ghép như One-point Crossover, Two-point Crossover, và Uniform Crossover, cùng với ảnh hưởng của chúng đến đặc điểm di truyền của quần thể.
6. Đột biến (Mutation)
    a. Vai trò: Đột biến giúp duy trì sự đa dạng di truyền trong quần thể bằng cách thay đổi ngẫu nhiên gen của cá thể.
    b. Phương pháp: Giới thiệu cách thức và tỷ lệ đột biến thường được sử dụng, bao gồm Bit-flip Mutation cho mã hóa nhị phân và các phương pháp khác cho số nguyên hoặc số thực.



```
START
Generate the initial population
Compute fitness
REPEAT
    Selection
    Crossover
    Mutation
    Compute fitness
UNTIL population has converged
STOP
```

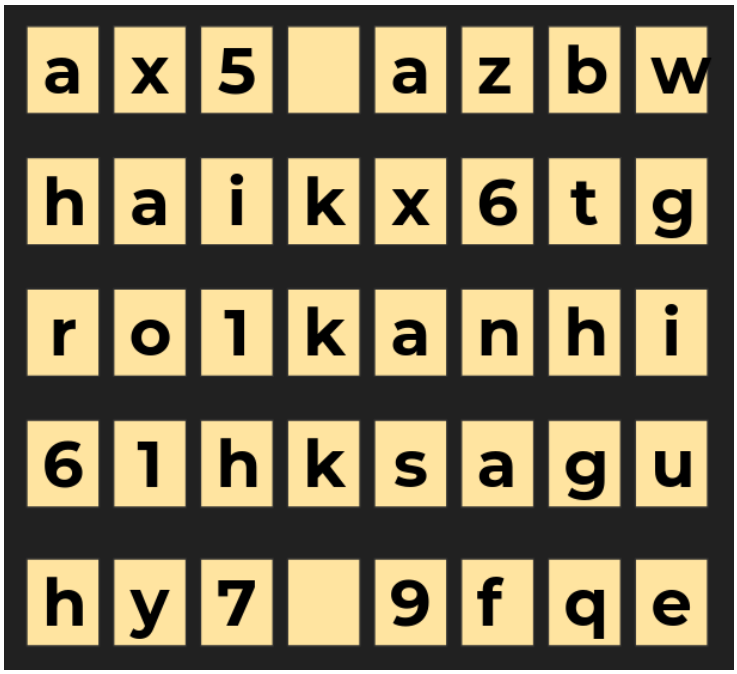
Hình 6 Mã giả

2.1.2 Bài toán điển hình

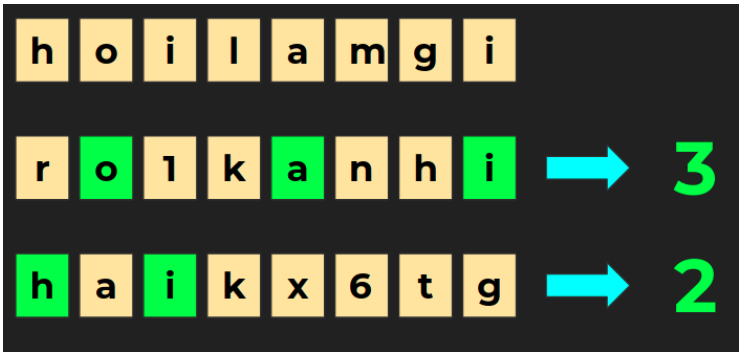
Bài toán Đoán Mật Khẩu (Password Guessing Problem): Trong bài toán này, mục tiêu là tìm ra một chuỗi ký tự (mật khẩu) bằng cách 'đoán' liên tục. GA sử dụng các toán tử di truyền như lai ghép và đột biến để tạo ra các thế hệ mới của các chuỗi ký tự, tiến tới mật khẩu đúng. Đây là một minh họa rất trực quan và hiệu quả của cách mà GA tối ưu hóa và tiến hóa dần tới giải pháp qua nhiều thế hệ.

Ví dụ minh họa: Xét bài toán Tìm mật khẩu, yêu cầu của bài toán như sau:

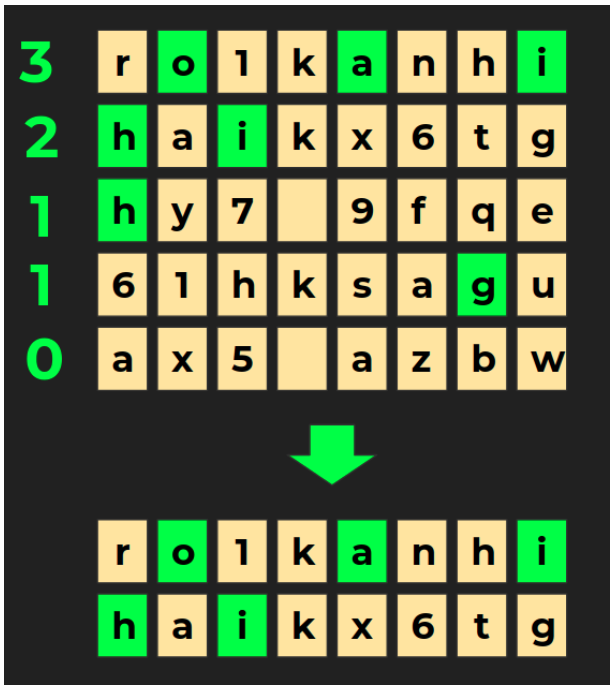
- Mật khẩu gồm 8 kí tự (bao gồm chữ cái, chữ số và khoảng trắng) - Ví dụ: hoilamgi.
- Mỗi lần thử, hệ thống sẽ báo về số lượng kí tự đúng với mật khẩu.
- Yêu cầu tìm ra chuỗi mật khẩu cho trước.



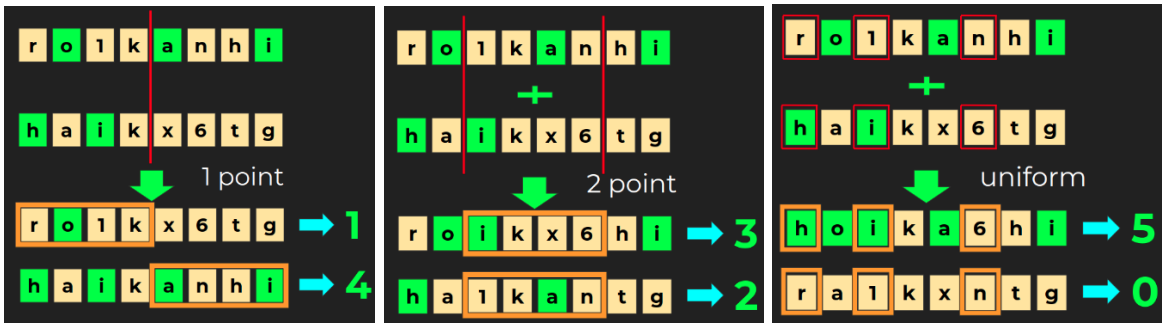
Hình 1: Khởi tạo quần thể bất kỳ với chiều dài mỗi mật khẩu là 8 ký tự



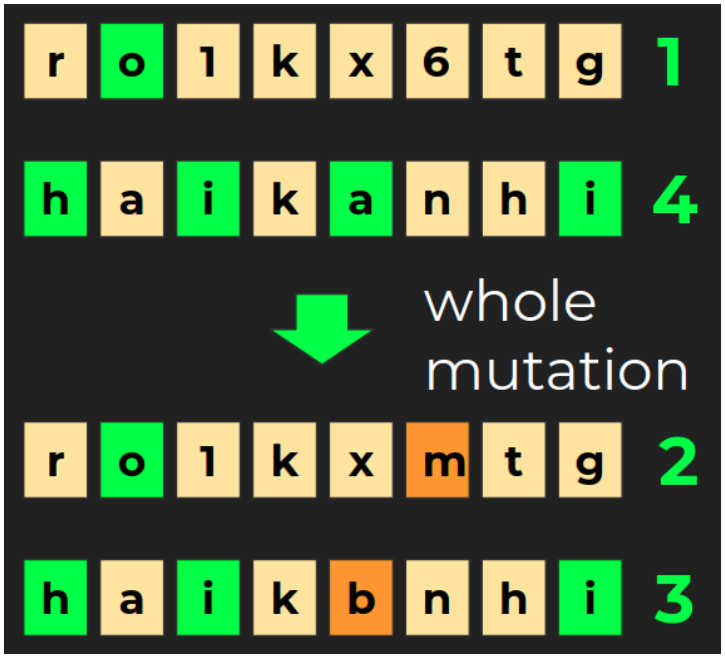
Hình 2: Đánh giá mật khẩu trong quần thể



Hình 3: lựa chọn cặp bố mẹ khỏe nhất sau khi đánh giá



Hình 4 a,b,c: Lai ghép cặp bố mẹ rồi đánh giá cặp con tương ứng



Hình 5: Đột biến con lai tạo sau khi lai ghép

2.1.3 Các biến thể của GA

1. Steady-State Genetic Algorithm (SSGA)
    - Đặc điểm: Trong SSGA, một hoặc một số cá thể mới được tạo ra và ngay lập tức thay thế một số cá thể kém thích nghi nhất trong quần thể, thay vì thay thế toàn bộ quần thể như trong GA tiêu chuẩn.
    - Ưu điểm: Giúp duy trì sự ổn định của quần thể và thường nhanh chóng hội tụ hơn so với GA tiêu chuẩn.
2. Elitist Genetic Algorithm
    - Đặc điểm: Phương pháp này bảo tồn một hoặc một số cá thể tốt nhất trong mỗi thế hệ (elites) không bị thay thế trong quá trình sinh sản.
    - Ưu điểm: Ngăn chặn mất mát của các giải pháp tốt nhất đã tìm thấy, đảm bảo rằng hiệu suất của quần thể không giảm qua các thế hệ.
3. Multi-Objective Genetic Algorithms (MOGA)
    - Đặc điểm: Được thiết kế để giải quyết các bài toán có nhiều mục tiêu tối ưu hóa đồng thời, mà không cần chuyển đổi chúng thành một mục tiêu đơn lẻ.
    - Ưu điểm: Có khả năng tìm ra một tập hợp các giải pháp tối ưu đa dạng, mỗi giải pháp tốt ở một mục tiêu nhất định, gọi là Pareto front.
4. Adaptive Genetic Algorithms
    - Đặc điểm: Tự động điều chỉnh các tham số của GA (như tỷ lệ đột biến, tỷ lệ lai ghép) dựa trên tiến trình tối ưu hóa.
    - Ưu điểm: Tăng khả năng thích ứng với bài toán cụ thể, cải thiện hiệu quả tìm kiếm và hội tụ.

5. Hybrid Genetic Algorithms
    - Đặc điểm: Kết hợp GA với các kỹ thuật tối ưu hóa khác (như Hill Climbing, Simulated Annealing, hoặc thậm chí là các thuật toán khác như Particle Swarm Optimization).
    - Ưu điểm: Tận dụng điểm mạnh của từng phương pháp để cải thiện hiệu quả tìm kiếm và khả năng hội tụ, đặc biệt hữu ích cho các bài toán phức tạp.
6. Parallel Genetic Algorithms
    - Đặc điểm: Phân tán quá trình GA trên nhiều máy tính hoặc nhiều luồng xử lý để thực hiện đồng thời.
    - Ưu điểm: Giảm thời gian tính toán đáng kể, đặc biệt quan trọng cho các bài toán lớn.
7. Real-Coded Genetic Algorithms
    - Đặc điểm: Sử dụng biểu diễn số thực thay vì chuỗi nhị phân để mã hóa các biến số.
    - Ưu điểm: Cải thiện độ chính xác và hiệu quả của GA trong các bài toán có biến số liên tục.
8. Constraint-Handling Techniques in GA
    - Đặc điểm: Áp dụng các kỹ thuật đặc biệt để xử lý các ràng buộc trong bài toán, giúp GA có thể áp dụng cho các bài toán tối ưu hóa có ràng buộc.
    - Ưu điểm: Mở rộng khả năng áp dụng của GA cho một loạt các bài toán thực tế có các ràng buộc phức tạp.

2.2 Thuật toán tối ưu hóa đàn kiến

- ACO được phát triển bởi Marco Dorigo vào đầu những năm 1990 trong khuôn khổ luận án tiến sĩ của ông tại Đại học Libre de Bruxelles, Bỉ. Thuật toán ban đầu được thiết kế để giải quyết bài toán người bán hàng (Travelling Salesman Problem - TSP), một trong những bài toán tối ưu hóa tổ hợp phổ biến và thách thức.
- Thuật toán ACO lấy cảm hứng từ hành vi tìm đường và xây dựng tổ của loài kiến trong tự nhiên. Khi kiếm ăn, kiến phát tán một chất hóa học gọi là pheromone trên đường đi của chúng. Các con kiến khác có thể cảm nhận được mùi pheromone và theo dõi nó để tìm thức ăn. Đường đi có nhiều pheromone hơn (tức là được nhiều kiến đi qua) trở nên hấp dẫn hơn, dẫn đến một quá trình tối ưu hóa tự nhiên qua đó đường đi tốt nhất dần được thiết lập.

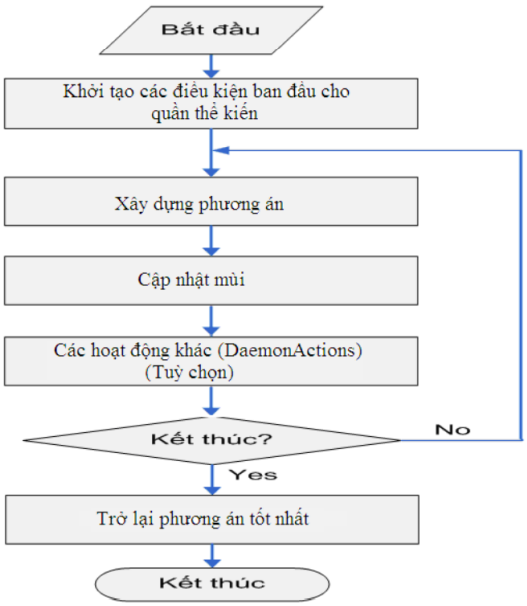
- 

2.1.1Các thành phần chính của ACO

1. Kiến (Ants)
    - Mô tả: Trong ACO, kiến được mô phỏng là các tác nhân giải quyết bài toán. Chúng bắt đầu từ một điểm, thường là điểm khởi đầu của bài toán, và dần dần xây dựng lộ trình bằng cách lựa chọn từng bước tiếp theo dựa trên pheromone và thông tin heuristic.

- Vai trò: Tác nhân của thuật toán, mỗi con kiến tương ứng với một lần chạy thuật toán hoặc một lần thử nghiệm giải pháp, và chúng cộng tác để khám phá không gian giải pháp.

2. Pheromone
   - Mô tả: Pheromone trong ACO là một biến số ảo mà kiến "để lại" trên lộ trình mà chúng đã đi qua. Lượng pheromone trên một lộ trình phản ánh mức độ thành công của lộ trình đó trong quá khứ, càng nhiều kiến đi qua và càng thành công thì lượng pheromone càng cao.
   - Vai trò: Đây là cơ chế chính để chia sẻ thông tin giữa các kiến, giúp thuật toán tập trung vào những lộ trình tốt hơn và tránh những lộ trình kém hiệu quả.
3. Heuristic Information
   - Mô tả: Thông tin heuristic là dữ liệu dựa trên bản chất của bài toán, chẳng hạn như khoảng cách giữa các điểm trong bài toán người bán hàng, hoặc chi phí liên quan đến một bước chuyển động trong bài toán tối ưu hóa.
   - Vai trò: Hỗ trợ kiến trong việc lựa chọn bước tiếp theo bằng cách cung cấp một "mức độ ưu tiên" tự nhiên dựa trên các tính toán đơn giản, giúp kết hợp sự khám phá ngẫu nhiên với việc tập trung vào giải pháp hiệu quả.
4. Hàm Mục tiêu (Objective Function)
   - Mô tả: Hàm mục tiêu là tiêu chuẩn để đánh giá chất lượng của một lộ trình hoặc giải pháp. Trong ACO, hàm mục tiêu thường được dùng để tính toán "mức độ thích hợp" của một lộ trình dựa trên tổng chi phí, khoảng cách, thời gian, hoặc các yếu tố tương tự.
   - Vai trò: Xác định mức độ cập nhật pheromone trên các lộ trình; lộ trình càng tối ưu theo hàm mục tiêu, lượng pheromone để lại càng nhiều, từ đó hướng dẫn các kiến trong các lần tìm kiếm sau tập trung vào những lộ trình đó.



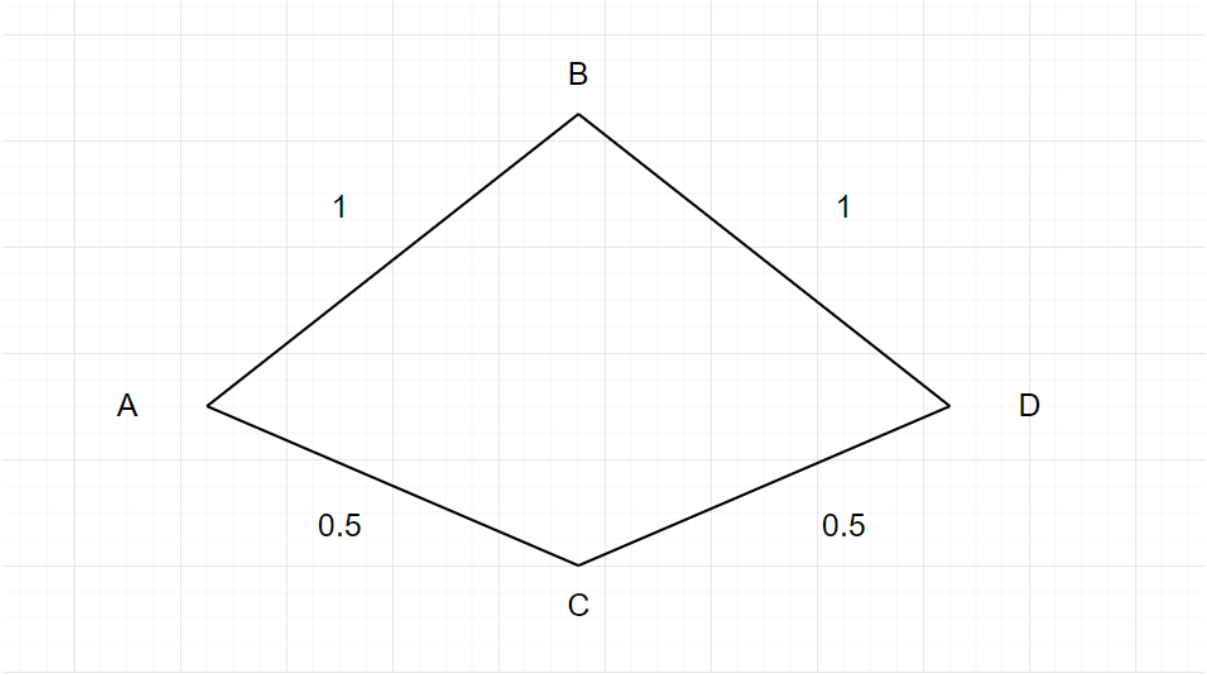
```
Procedure Thuật toán ACO;
Begin
   Khởi tạo tham số, ma trận mùi, khởi tạo m con kiến;
   repeat
      for k = 1 to m do
         Kiến k xây dựng lời giải;
      end-for
      Cập nhật mùi;
      Cập nhật lời giải tốt nhất;
   until (Điều kiện kết thúc);
   Đưa ra lời giải tốt nhất;
End;
```

Hình 7 Mã giả

2.2.2 Bài toán điển hình

Bài toán tìm đường đi ngắn nhất giữa điểm bắt đầu tới điểm đích là 1 bài toán điển hình của ACO nhằm tối ưu đường đi tới đích một cách ngắn nhất nhanh nhất.

Ví dụ: Cho 4 đỉnh A, B, C, D với AB=BD = 1 , AC=CD=0.5 đơn vị khoảng cách tổ kiến bắt đầu từ A và thức ăn tại D. Tìm đường đi ngắn nhất



Khởi tạo các giá trị

- $\tau_{A,B}, \tau_{A,C}$: Mức pheromone ban đầu trên các cạnh A-B và A-C. Giả sử ban đầu là 0.1 cho cả hai cạnh
- $\eta_{A,B}, \eta_{A,C}$: Độ thuận lợi của cạnh, tính bằng nghịch đảo của khoảng cách từ A đến B và từ A đến C. với khảng cách AB = 1 và AC = 0.5, ta có $\eta_{A,B}$ = 1/1 = 1 và $\eta_{A,C}$ = 1/0.5 = 2
- α, β: Các tham số điều chỉnh ảnh hưởng của pheromone (α) và thông tin heuristic (β). Giả sử $\alpha = 1,\ \beta = 2$.

Công thức tính xác suất:

Xác suất $P_{i,j}$ để kiến chọn đi từ điểm i đến j được tính bằng công thức:

$$P_{i,j} = \frac{(\tau_{i,j})^{\alpha}.(\eta_{i,j})^{\beta}}{\sum_{k \in adj(i)}(\tau_{i,k})^{\alpha}.(\eta_{i,k})^{\beta}}$$

Xác suất từ A đến B ($P_{A,B}$):

$$P_{A,B} = \frac{(0.1)^1.(1)^2}{(0.1)^1.(1)^2 + (0.1)^1.(2)^2} = \frac{0.1}{0.5} = 0.2$$

Giải thích:

- $\tau_{A,B}, \tau_{A,C} = 0.1$
- $\eta_{A,B} = 1, \eta_{A,C} = 2$
- $\alpha = 1, \beta = 2$

Xác suất từ A đến B ($P_{A,B}$):

$$P_{A,C} = \frac{(0.1)^1.(2)^2}{(0.1)^1.(1)^2 + (0.1)^1.(2)^2} = \frac{0.4}{0.5} = 0.8$$

=> Từ đó kiến sẽ chọn đường đi có xác suất cao nhất để đi. Nhưng vẫn còn có trong số các con kiến đi đường có xác suất ít để đi và kiểm tra đường đi khác, vì đâu chắc tổng đường đi chứa đường có xác suất nhỏ nhất đang xét nó nhỏ hơn tổng đường đi của các đường chứa đường có xác xuất lớn hơn.

2.2.3 Các biến thể của ACO

1. Ant System (AS)
    - Mô tả: AS là phiên bản ban đầu của ACO. Trong AS, tất cả kiến được cho là cập nhật pheromone dựa trên chất lượng của lộ trình mà chúng đã tìm thấy.
    - Ứng dụng: Phù hợp cho các bài toán có không gian tìm kiếm không quá lớn, nơi mà sự thăm dò có thể được thực hiện một cách có hệ thống.
2. Ant Colony System (ACS)
    - Mô tả: Trong ACS, chỉ những kiến tìm thấy lộ trình tốt nhất mới được cập nhật pheromone, điều này giúp tập trung vào việc khai thác các lộ trình tối ưu.
    - Ứng dụng: Đặc biệt hiệu quả trong các bài toán đòi hỏi khả năng khai thác nhanh chóng các giải pháp tốt.
3. Max-Min Ant System (MMAS)
    - Mô tả: MMAS đặt giới hạn trên và dưới cho lượng pheromone, ngăn chặn sự thống trị của bất kỳ lộ trình nào để đảm bảo sự khám phá đầy đủ hơn của không gian tìm kiếm.
    - Ứng dụng: Phù hợp cho các bài toán có nguy cơ cao bị mắc kẹt ở cực tiểu địa phương.
4. Ant-Q
    - Mô tả: Là sự kết hợp giữa ACO và Q-learning, một kỹ thuật học tăng cường. Trong Ant-Q, kiến cập nhật giá trị của pheromone dựa trên một quá trình học có thưởng thức.
    - Ứng dụng: Hiệu quả trong các môi trường động, nơi các kiến cần thích ứng với sự thay đổi của bài toán theo thời gian.
5. Rank-based Ant System (RAS)
    - Mô tả: Chỉ những kiến tìm thấy các lộ trình tốt nhất mới được phép cập nhật pheromone, nhưng dựa trên xếp hạng thay vì chỉ có một kiến tốt nhất.
    - Ứng dụng: Cung cấp sự cân bằng tốt hơn giữa khám phá và khai thác, thích hợp cho các bài toán có không gian tìm kiếm rộng.
6. Hypercube Framework for Ant Colony Optimization (HACO)
    - Mô tả: Trong HACO, pheromone được xử lý trong một không gian nhiều chiều, cho phép thể hiện phức tạp hơn các mối quan hệ giữa các thành phần.
    - Ứng dụng: Đặc biệt hữu ích trong các bài toán tối ưu hóa đa mục tiêu hoặc khi các quyết định có mối liên hệ phức tạp với nhau.

Phần 3: Ứng dụng GA và ACO vào bài toán người bán hàng (Travelling Salesman Problem - TSP)

1. Ý tưởng thực hiện

1.1 Thuật toán Kiến Tìm Đường (Ant Colony Optimization - ACO)

Ý tưởng cơ bản: Mô phỏng hành vi tìm đường và xây dựng đường đi của kiến trong tự nhiên, dựa trên việc tích lũy pheromone trên đường đi để hướng đến giải pháp tối ưu.

Cách tiếp cận:

- Khởi tạo: Đặt một lượng pheromone ban đầu nhỏ trên tất cả các cạnh nối giữa các thành phố.
- Xây dựng lộ trình: Mỗi con kiến sẽ chọn lộ trình tiếp theo dựa trên xác suất, ưu tiên các cạnh có nhiều pheromone và khoảng cách ngắn.
- Cập nhật Pheromone: Sau khi tất cả kiến hoàn thành lộ trình của mình, giảm pheromone trên tất cả các cạnh và tăng cường pheromone trên lộ trình tối ưu nhất mà kiến đã đi qua.

Lặp lại: Quá trình này được lặp lại cho đến khi đạt đến một điều kiện dừng, chẳng hạn như số lần lặp tối đa hoặc khi giải pháp không còn cải thiện nữa.

1.2 Thuật toán Di truyền (Genetic Algorithms - GA)

Ý tưởng cơ bản: Mô phỏng quá trình tiến hóa sinh học, bao gồm các phép toán như chọn lọc tự nhiên, lai ghép và đột biến để tìm ra giải pháp tối ưu.

Cách tiếp cận:

- Khởi tạo: Tạo một quần thể ban đầu gồm nhiều cá thể, mỗi cá thể là một giải pháp tiềm năng (lộ trình qua các thành phố).
- Đánh giá: Tính toán độ thích nghi của mỗi cá thể, thường là tổng khoảng cách của lộ trình.
- Chọn lọc: Chọn các cá thể có độ thích nghi cao để sinh sản.
- Lai ghép và Đột biến: Tạo ra thế hệ mới thông qua các phép lai ghép (kết hợp đặc điểm của hai cá thể cha mẹ) và đột biến (thay đổi ngẫu nhiên một phần của giải pháp).

Lặp lại: Quá trình này được lặp đi lặp lại cho đến khi đạt được giải pháp mong muốn hoặc khi đã thực hiện đủ số thế hệ.

2. Thực nghiệm

3. Kết luận và đánh giá

3.1 Đánh giá hiệu quả

Thuật toán di truyền (GA)

Ưu điểm: GA tốt trong việc tìm kiếm trên không gian lớn và có khả năng thoát khỏi các điểm tối ưu cục bộ nhờ vào cơ chế đột biến và lai ghép.

Nhược điểm: Kết quả tối ưu phụ thuộc nhiều vào các tham số như tỷ lệ đột biến và cách thức lai ghép. Đôi khi, việc tinh chỉnh các tham số này để đạt hiệu quả tối ưu có thể rất phức tạp và mất thời gian.

Thuật toán kiến (ACO)

Ưu điểm: ACO rất mạnh trong việc tìm kiếm đường đi dựa trên cơ chế pheromone, cho phép chia sẻ thông tin giữa các cá thể, từ đó hướng tới giải pháp tốt chung.

Nhược điểm: ACO có thể mất nhiều thời gian hơn để hội tụ, đặc biệt là khi kích thước bài toán lớn, do nó phụ thuộc vào sự phân bố pheromone và cần nhiều vòng lặp để điều chỉnh pheromone đạt được mức tối ưu.

3.2 Phức tạp thuật toán

- GA: Phức tạp chủ yếu ở các bước lai ghép và đột biến. Việc quản lý quần thể đòi hỏi kỹ thuật lập trình tốt để đảm bảo hiệu quả.
- ACO: Phức tạp ở phần tính toán pheromone và cách thức các kiến chọn lựa đường đi. Đòi hỏi sự cân bằng giữa các tham số như suy giảm pheromone và hệ số heuristic.

3.3 Tính mềm dẻo và khả năng thích ứng

- GA: Có khả năng thích ứng cao với các bài toán khác nhau bằng cách thay đổi cách thức lai ghép và đột biến.
- ACO: Tuy nhiên, ACO cũng có thể thích ứng với nhiều loại bài toán tối ưu hóa khác nhau, nhưng cần điều chỉnh cẩn thận các tham số liên quan đến pheromone.

3.4 Tính chất hội tụ

- GA: Có thể hội tụ nhanh chóng nếu quần thể đa dạng và kích thước quần thể đủ lớn.
- ACO: Hội tụ dựa trên sự tích lũy và bay hơi pheromone, có thể chậm hơn GA nhưng đôi khi lại mang lại giải pháp tốt hơn do tính toán kỹ lưỡng hơn.

Tài liệu tham khảo

1 Tài liệu chuyên tin học quyển 3

2 https://www.geeksforgeeks.org/genetic-algorithms/?ref=header_search
3 https://www.geeksforgeeks.org/encoding-methods-in-genetic-algorithm/?ref=header_search
4 https://www.geeksforgeeks.org/project-idea-genetic-algorithms-for-graph-colouring/?ref=header_search'
5 https://intapi.sciendo.com/pdf/10.2478/v10238-012-0039-2
6 https://core.ac.uk/download/pdf/32452919.pdf
7 https://www.linkedin.com/pulse/use-genetic-algorithms-planning-scheduling-projects-well-suleiman
8 https://nerophung.github.io/2020/05/28/genetic-algorithm#thu%E1%BA%ADt-to%C3%A1n-di-truy%E1%BB%81n
9 https://medium.datadriveninvestor.com/genetic-algorithm-made-intuitive-with-natural-selection-and-python-project-from-scratch-3462f7793a3f
10 https://www.youtube.com/watch?v=fxCHAkYAjis&list=PLALZwazFybKKeKPXDP4EjrDLNW436xiLI&index=2
11 https://www.youtube.com/watch?v=MGY3Vf_M4Qs
12 https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3
13 https://www.geeksforgeeks.org/introduction-to-ant-colony-optimization/
14 https://www.geeksforgeeks.org/introduction-to-ant-colony-optimization/?ref=header_search
15 https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3