**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TPHCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**\_oOo\_**

**BÁO CÁO MÔN HỌC**

**TRÍ TUỆ NHÂN TẠO**

**NGHIÊN CỨU VỀ GIẢI THUẬT DUY TRUYỀN VÀ BÀI TOÁN NGƯỜI BÁN HÀNG**

Ngành : CÔNG NGHỆ THÔNG TIN

Môn học : TRÍ TUỆ NHÂN TẠO

Sinh viên thực hiện :

TP.Hồ Chí Minh,2021

**LỜI NÓI ĐẦU**

Trong ngành khoa học máy tính, tìm kiếm lời giải tối ưu cho bài toán là vấn đề được các nhà khoa học máy tính đặc biệt quan tâm.

Mục đích chính của các thuật toán tìm kiếm lời giải là tìm ra lời giải pháp tối ưu nhất cho bài toán trong thời gian ngắn nhất. Các thuật toán như tìm kiếm không có thông tin (tìm kiếm trên danh sách, trên cây hoặc đồ thị) sử dụng phương pháp đơn giản nhất và trực quan nhất hoặc các thuật toán tìm kiếm thông tin sử dụng Heuritics để áp dụng các tri thức về cấu trúc không gian tìm kiếm nhằm giảm thời gian cần thiết cho việc tìm kiếm được sử dụng nhiều nhưng chỉ với không gian tìm kiếm nhỏ và không hiệu quả khi tìm kiếm trong không gian lớn.

Tuy nhiên trong thực tiễn có rất nhiều bài toán tối ưu với không gian tìm kiếm rất lớn cần giải quyết. Vì vậy, việc đòi hỏi thuật giải chất lượng cao và sử dụng kỹ thuật trí tuệ nhân tạo đặc biệt cần thiết khi giải quyết các bài toán có không gian tìm kiếm lớn. Thuật giải di truyền(Genetic Algorithms) là một trong những kỹ thuật tìm kiếm lời giải tối ưu đã đáp ứng được yêu cầu của nhiều bài toán và ứng dụng.

Thuật toán di truyền cùng với logic mờ được ứng dụng rất rộng rãi trong các lĩnh vực phức tạp. Thuật toán di truyền kết hợp với logic mờ chứng tỏ được hiệu quả của nó trong các vấn đề khó có thể giải quyết bằng phương pháp thông thường hay các phương pháp cổ điển, nhất là các bài toán cần có sự lượng giá, đánh giá sự tối ưu của kết quả thu được. Chính vì vậy, thuât giải di truyền đã trở thành đề tài nghiên cứu thú vị và đem đến nhiều ứng dụng trong đời sống thực tiễn.

Ngày nay,thuật toán di truyền được ứng dụng khá nhiều trong các lĩnh vực như khoa học, kinh doanh và giải trí. Đầu tiên phải kể đến là các bài toán tối ưu bao gồm tối ưu số và tối ưu tổ hợp đã sử dụng GAS để tìm lời giải như bài toán người bán hàng (Travelling Salesman Problem – TSP). Ứng dụng kế tiếp của GAs là thiết kế và điều khiển robo. Hầu hết các nước có ngành CNTT phát triển đã và đang rất quan tâm đến ĩnh vực thiết kế robo nhằm giúp con người tiết kiệm sức lao động và giải phóng con người khỏi những công việc nguy hiểm, đặc biệt hiện nay cuộc thi “Robocon” Châu Á\_Thái Bình Dương được các nước trong khu vực rất quan tâm. Ngoài phần cơ, để robo có thể tiến hành các hoạt động đơn giản nhất như đi, đứng… thì robo cần phải trang bị chương trình được lập trình dựa trên các thuật toán và ngôn ngữ thích hợp. Nhờ vào lịch trình được cài đặt cùng với một trí tuệ nhân tạo nhân tạo…, robo có thể định hướng thực hiện các hoạt động như con người. Tuy nhiên, việc tìm kiếm lời giải tốt nhất cho hành động của robo không phải là đơn giản. Theo các nhà khoa học máy tính, thuật giải di truyền là một trong những thuật toán tối ưu giúp robo vạch lộ trình khi di chuyển.

Bài toán người bán hàng là một trong những bài toán được nghiên cứu sâu nhất trong lĩnh vực tối ưu hóa. Với lý do trên, em chọn đề tài “Thuật toán di truyền và ứng dụng bài toán người bán hàng”.Báo cáo này sẽ trình bày 1 hướng tiếp cận giải quyết bài toán bán hàng sử dụng giải thuật di truyền.

**CHƯƠNG 1: TỔNG QUAN VỀ GIẢI THUẬT DI TRUYỀN**

1. **Khái niệm**

**Giải thuật di truyền (GAs-Genetic Algorithms)** là một kỹ thuật của khoa học máy tính nhằm tìm kiếm giải pháp thích hợp cho các bài toán tối ưu tổ hợp (combinatorial optimization), là một phân ngành của giải thuật tiến hóa, vận dụng các nguyên lý của tiến hóa như: di truyền, đột biến, chọn lọc tự nhiên, và trao đổi chéo. Nó sử dụng ngôn ngữ máy tính để mô phỏng quá trình tiến hoá của một tập hợp những đại diện trừu tượng (gọi là những nhiễm sắc thể), của các giải pháp có thể (gọi là những cá thể) cho bài toán tối ưu hóa vấn đề. Tập hợp này sẽ tiến triển theo hướng chọn lọc những giải pháp tốt hơn.

GAs cũng như các thuật toán tiến hoá, đều được hình thành dựa trên một quan niệm được coi là một tiên đề phù hợp với thực tế khách quan. Đó là quan niệm "Quá trình tiến hoá tự nhiên là quá trình hoàn hảo nhất, hợp lý nhất và tự nó đã mang tính tối ưu". Quá trình tiến hoá thể hiện tính tối ưu ở chỗ thế hệ sau bao giờ cũng tốt hơn thế hệ trước.

Ngày nay, GAs càng trở nên quan trọng, đặc biệt là trong lĩnh vực tối ưu hoá, một lĩnh vực có nhiều bài toán thú vị, được ứng dụng nhiều trong thực tiễn nhưng thường khó và chưa có phương pháp hiệu quả để giải quyết.

1. **Lịch sử hình thành và phát triển**

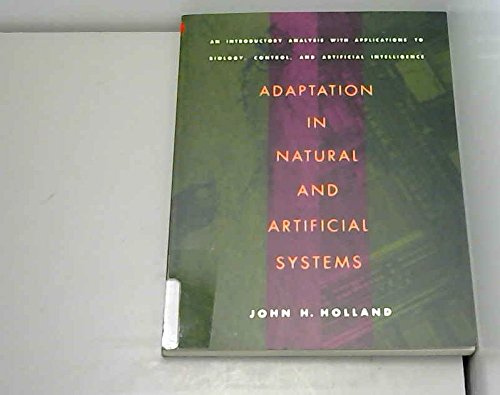
Thuật toán di truyền (Genetic algorithms - GAs) đã được phát minh bởi John Holland trong những năm 1960; sau đó được tiếp tục phát triển bởi sinh viên và đồng nghiệp của ông tại Đại học Michigan trong thập niên 1960 và thập niên 1970. Ngược lại với các chiến lược phát triển và lập trình tiến hóa, mục tiêu ban đầu của Holland không để thiết kế các thuật toán để giải quyết các vấn đề cụ thể, mà là để chính thức nghiên cứu các hiện tượng của sự thích nghi khi nó xảy ra trong tự nhiên và phát triển những cách thức mà các cơ chế thích ứng tự nhiên có thể được đem vào vào hệ thống máy tính.

1975 cuốn sách *Adaptation in Natural and Artificial Systems*của Holland đã trình bày các thuật toán di truyền như là một sự trừu tượng của sự tiến hóa sinh học và đã đưa ra một khuôn khổ lý thuyết cho thích ứng theo GA. GA của Holland là một phương pháp để di chuyển từ một dân số "nhiễm sắc thể" (ví dụ, chuỗi của những số 1 và số 0, hoặc "bits") cho một dân số mới bằng cách sử dụng một loại "chọn lọc tự nhiên" cùng với các  hoạt động di truyền như lai chéo, đột biến, và đảo ngược. Mỗi nhiễm sắc thể bao gồm "Genetics" (ví dụ, bits), mỗi gen là một thể hiện của một "allele" đặc biệt (ví dụ, 0 hoặc 1). Thuật toán sẽ lựa chọn chọn những nhiễm sắc thể trong dân số sẽ được cho phép để tái sản xuất và sinh sản nhiều hơn so với những người ít phù hợp. Trao đổi chéo (Crossover) các thành phần con của hai nhiễm sắc thể bắt chước tái tổ hợp sinh học giữa hai đơn nhiễm sắc thể ("đơn bội") sinh vật. Đột biến thay đổi ngẫu nhiên (mutation randomly) các giá trị alen của một số địa điểm trong các nhiễm sắc thể; và đảo ngược thứ tự (Inversion Reverses) của một phần tiếp giáp của các nhiễm sắc thể, do đó sắp xếp lại thứ tự các gen.

Thuật toán dựa trên dân số của Holland với các với hoạt động lai ghép, đảo ngược, và đột biến là một sự đổi mới lớn và là nguồn cảm hứng cho việc giải quyết các vấn đề về tính toán.



Hình 1: Nhà khoa học John Henry Holand (1929-2015)



Hình 2: Cuốn sách Adaptation in Natural and Artificial Systems của nhà khoa học John Henry Holland

Thông thường, các vấn đề này đòi hỏi tìm kiếm một tập liên tục thay đổi các khả năng với số lượng lớn các khả năng để thích nghi cho điều kiện môi trường thay đổi. Các giải pháp. tìm kiếm như vậy thường yêu cầu sử dụng hiệu quả của xử lý song song, trong đó có nhiều khả năng khác nhau được khám phá cùng một lúc một cách hiệu quả để tạo ra thay đổi hàng triệu loài song song. Có thể nói, nhìn từ một mức độ cao các "nguyên tắc" của tiến hóa là khá đơn giản: các loài tiến hóa bằng phương tiện của biến ngẫu nhiên (thông qua đột biến, tái tổ hợp, và khai khác), tiếp theo là chọn lọc tự nhiên trong đó có xu hướng các thích nghi để tồn tại và tái sản xuất; do đó truyền cho các thế hệ tương lai tạo ra sự đa dạng bất thường và phức tạp như chúng ta thấy trong sinh quyển ngày nay.

**CHƯƠNG 2: ĐẶC ĐIỂM GIẢI THUẬT DI TRUYỀN**

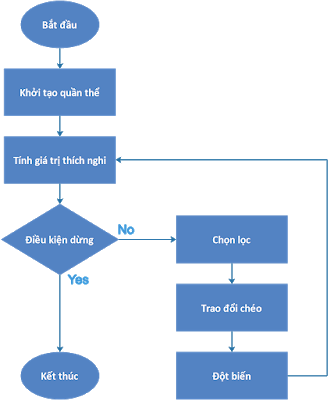
1. **Tính chất**

GAs là kỹ thuật chung, giúp giải quyết vấn đề bằng cách mô phỏng sự tiến hóa của con người hay của sinh vật nói chung (dựa trên thuyết tiến hóa muôn loài của Darwin), trong điều kiện qui định sẵn của môi trường. Mục tiêu của GAs không nhằm đưa ra lời giải chính xác tối ưu mà là đưa ra lời giải tương đối tối ưu.

Một cá thể trong GAs sẽ biểu diễn một giải pháp của bài toán. Tuy nhiên, không giống với trong tự nhiên là một cá thể có nhiều nhiễm sắc thể (NST) mà để giới hạn trong GAs, ta quan niệm một cá thể có một NST. Do đó, khái niệm cá thể và NST trong GAs coi như là tương đương.

Một NST được tạo thành từ nhiều gen, mỗi gen có thể có các giá trị khác nhau để quy định một tình trạng nào đó. Trong GAs, một gen được coi như một phần tử trong chuỗi NST.

1. **Các bước cơ bản**



Hình 3: Sơ đồ tổng quan giải thuật di truyền

Như trên hình 3, ta thấy giải thuật di truyền đơn giản được thực hiện qua các bước sau:

Giải thuật sẽ được thực hiện qua các bước sau:

* Bắt đầu: Nhận các tham số cho thuật toán.
* Khởi tạo quần thể: Sinh ra ngẫu nhiên một quần thể gồm n cá thể (trong đó n là lời giải cho bài toán).
* Tính giá trị thích nghi: Ước lượng độ thích nghi của mỗi cá thể.
* Điều kiện dừng: Kiểm tra điều kiện để kết thúc giải thuật.
* Chọn lọc: Chọn hai cá thể bố mẹ từ quần thể cũ theo độ thích nghi của chúng (cá thể có độ thích nghi càng cao thì càng có nhiều khả năng được chọn).
* Trao đổi chéo: Với một xác suất được chọn, trao đổi chéo hai cá thể bố mẹ để tạo ra một cá thể mới.
* Đột biến: Với một xác suất đột biến được chọn, biến đổi cá thể mới.
* Chọn kết quá: Nếu thỏa mãn điều kiện dừng thì giải thuật kết thúc và chọn được lời giải tốt nhất trong quần thể hiện tại.

Ta có thể thấy rằng, khi *Điều kiện dừng* chưa được thỏa mán, quần thể mới sẽ liên tục được tạo ra bằng cách lặp lại 3 bước Chọn lọc, Trao đổi chéo và Đột biến

GAs có 2 điều kiện dừng cơ bản:

* Dựa trên cấu trúc nhiễm sắc thể, kiểm soát số gene được hội tụ, nếu số gene được hội tụ tại một điểm hoặc vượt quá điểm đó thì giải thuật kết thúc.
* Dựa trên ý nghĩa đặc biệt của nhiễm sắc thể, đo sự thay đổi của giải thuật sau mỗi thế hệ, nếu thay đổi này nhỏ hơn một hằng số xác định thì giải thuật kết thúc.

1. **Thuộc tính của sơ đồ giải thuật di truyền**

Các sơ đồ khác nhau có đặc trưng khác nhau. Các đặc trưng này thể hiện qua hai thuộc tính quan trọng: bậc và chiều dài xác định.

Bậc của sơ đồ S (ký hiệu o(S)) là tổng số vị trí 0, 1 có trong sơ đồ. Đây là các vị trí cố định (không phải vị trí của các ký tự đại diện) trong sơ đồ. Bậc có thể xác định bằng cách lấy chiều dài của chuỗi trừ đi số ký tự đại diện.

Trong sơ đồ S = (\* \* 1 0 \* 1 \*) có bậc o(S) = 7 - 4 = 3;

Chiều dài xác định của sơ đồ S (ký hiệu là δ(S)) là khoảng cách giữa 2 vị trí cố định ở đầu và cuối. Chiều dài của sơ đồ xác định độ nén thông tin chứa trong sơ đồ đó. Trong ví dụ trên δ(S) = 6-3 =3. Như vậy, nếu sơ đồ chỉ có một vị trí cố định thì chiều dài xác định của sơ đồ sẽ bằng 0.

Chiều dài của sơ đồ giúp ta tính xác suất tồn tại của sơ đồ do ảnh hưởng của ghép chéo.

Đặc điểm hội tụ của giải thuật di truyền.

Khi áp dụng giải thuật GAs cho các vấn đề thực tế thường rất khó khăn. Lý do:

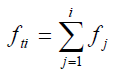
* Cách biểu diễn nhiễm sắc thể có thể tạo ra không tìm kiếm khác với không gian thực của bài toán;
* Số bước lặp, khi cài đặt thường không xác định trước;
* Kích thước quần thể thường có giới hạn.

Trong một số trường hợp, GAs không thể tìm được lời giải tối ưu. Lý do, GAs hội tụ sớm về lời giải tối ưu cục bộ. Hội tụ sớm là vấn đề của giải thuật di truyền cũng như các giải thuật tối ưu khác. Nếu hội tụ xảy ra quá nhanh thì các thông tin đáng tin cậy đang phát triển trong quần thể thường bị bỏ qua. Nguyên nhân của sự hội tụ sớm liên quan tới hai vấn đề:

* Quy mô và loại sai số do cơ chế tạo mẫu;
* Bản chất của hàm mục tiêu.

1. **Các toán tử di truyền**
   1. **Toán tử sinh sản**

Toán tử sinh sản gồm hai quá trình: quá trình sinh sản (phép tái sinh), quá trình chọn lọc (phép chọn).

* + 1. **Phép tái sinh**
* Phép tái sinh là quá trình các nhiễm sắc thể được sao chép trên cơ sở độ thích nghi. Độ thích nghi là một hàm được gán giá trị thực, tương ứng với mỗi nhiễm sắc thể trong quần thể. Quá trình này, được mô tả như sau:
* Xác định độ thích nghi của từng nhiễm sắc thể trong quần thể ở thế hệ thứ t, lập bảng cộng dồn các giá trị thích nghi (theo thứ tự gán cho từng nhiễm sắc thể). Giả sử, quần thể có n cá thể. Gọi độ thích nghi của nhiễm sắc thể i tương ứng là *fi*tổng cộng dồn thứ i là fti được xác định bởi:
* 
* Gọi *Fn*là tổng độ thích nghi của toàn quần thể. Chọn một số ngẫu nhiên *f*trong khoảng từ 0 tới *Fn*. Chọn cá thể thứ k đầu tiên thoả mãn *f ≥ ftk* đưa vào quần thể mới.
  + 1. **Phép chọn**

Phép chọn là quá trình loại bỏ các nhiễm sắc thể kém thích nghi trong quần thể. Quá trình này được mô tả như sau:

* Sắp xếp quần thể theo thứ tự mức độ thích nghi giảm dần.
* Loại bỏ các nhiễm sắc thể ở cuối dãy. Giữ lại n cá thể tốt nhất.
  1. **Toán tử ghép chéo**

Ghép chéo là quá trình tạo nhiễm sắc thể mới trên cơ sở các nhiễm sắc thể cha-mẹ bằng cách ghép một đoạn trên nhiễm sắc thể cha-mẹ với nhau. Toán tử ghép chéo được gán với một xác suất *p c* . Quá trình được mô tả như sau:

Chọn ngẫu nhiên một cặp nhiễm sắc thể (cha-mẹ) trong quần thể. Giả sử, nhiễm sắc thể cha-mẹ có cùng độ dài m.

Tạo một số ngẫu nhiên trong khoảng từ 1 tới m-1 (gọi là điểm ghép chéo). Điểm ghép chéo chia nhiễm sắc thể cha-mẹ thành hai chuỗi con có độ dài *m1, m2*. Hai chuỗi con mới được tạo thành là: *m11+ m22 và m21+m12.*

Đưa hai nhiễm sắc thể mới vào quần thể.

* 1. **Toán tử đột biến**

Đột biến là hiện tượng nhiễm sắc thể con mang một số đặc tính không có trong mã di truyền của cha-mẹ.

• Chọn ngẫu nhiên một nhiễm sắc thể trong quần thể;

• Tạo một số ngẫu nhiên k trong khoảng từ 1 tới m,*1 ≤ k ≤ m*;

• Thay đổi bit thứ k. Đưa nhiễm sắc thể này vào quần thể để tham gia quá trình tiến hoá ở thế hệ tiếp theo

1. **Các tham số của giải thuật di truyền**

Giải thuật di truyền có các tham số quan trọng như kích thước quần thể (popsize), xác suất lai ghép (pc), xác suất đột biến (pm). Việc lựa chọn các tham số phù hợp sẽ tăng tính hiệu quả của giải thuật.

Trong các tham số trên thì popsize là quan trọng nhất, nếu chọn kích thước quần thể quá nhỏ thì tính ña dạng của quần thể bị hạn chế và ảnh hưởng ñến kết quả còn nếu quá lớn sẽ làm hao phí tài nguyên của máy tính và làm chậm quá trình tiến hóa.

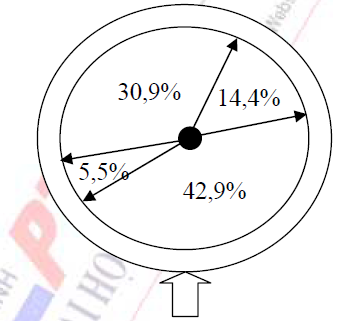
1. **Cơ chế tạo mẫu**

Có hai vấn đề quan trọng trong tiến trình tiến hoá của giải thuật di truyền là: tính đa dạng của quần thể và áp lực chọn lọc. Hai yếu tố này liên quan mật thiết với nhau: khi tăng áp lực chọn lọc thì tính đa dạng của quần thể sẽ giảm và ngược lại. Nói cách khác, áp lực hội tụ mạnh sẽ dẫn tới sự hội tụ sớm của giải thuật. Nhưng nếu áp lực chọn lọc yếu có thể làm cho tìm kiếm thành vô hiệu. Như vậy, cần thoả hiệp hai vấn đề. Hiện nay, các phương pháp đưa ra đều có khuynh hướng dể đạt tới mục đích này.

Năm 1975 DeJong đã xem xét một số biến thể của chọn lọc đơn giản bằng cách đưa ra: mô hình phát triển ưu tú, mô hình giá trị mong đợi và mô hình nhân tố tập trung.

Năm 1981 Brindle xem xét một số biến thể khác như: tạo mẫu tất định, tạo mẫu hỗn loạn, tạo mẫu hỗn loạn phần dư không thay thế, đấu tranh hỗn loạn, tạo mẫu hỗn loạn phần dư có thay thế.

Năm 1987 Baker nghiên cứu phương pháp tạo mẫu không gian hỗn loạn. Phương pháp này dùng cách “quay” bánh xe định tỷ lệ trước để thực hiện chọn lọc. Bánh xe này được thiết kế theo chuẩn, quay với số khoảng chia đều theo kích thước quần thể.



Hình 4: Tỷ lệ thích nghi của các nhiễm sắc thể trên bánh xe Roulette

Người ta thực hiện việc sinh sản bằng cách quay bánh xe Roulette với số lần bằng số nhiễm sắc thể trên bánh xe Roulette. Đối với bài toán này số lần quay bánh xe Roulette là 4. Nhiễm sắc thể 1 có giá trị thích nghi là 169, tương ứng 14,4 % tổng độ thích nghi. Như vậy, nhiễm sắc thể 1 chiếm 14.4% trên bánh xe Roulette. Mỗi lần quay nhiễm sắc thể 1 sẽ chiếm khe với giá trị 0,144.

Khi yêu cầu sinh ra 1 thế hệ mới, một vòng quay của bánh xe Roulette được đánh trọng số phù hợp sẽ chọn ra một cá thể để sinh sản. Bằng cách này, những nhiễm sắc thể có độ thích nghi cao sẽ có cơ hội được chọn lớn. Như vậy, sẽ có 1 số lượng con cháu lớn trong các thế hệ kế tiếp

**CHƯƠNG 3: THỰC THI VÀ ỨNG DỤNG GIẢI THUẬT DI TRUYỀN**

1. **Nguyên lý hoạt động**

Nền tảng lý thuyết của GAs dựa trên biểu diễn chuỗi nhị phân và lý thuyết sơ đồ. Một sơ đồ là một chuỗi, dài bằng chuỗi nhiễm sắc thể, các thành phần của nó có thể nhận một trong các giá trị của tập ký tự biểu diễn gen hoặc một ký tự đại diện “\*”.

Sơ đồ biểu diễn một không gian con của không gian tìm kiếm. Không gian con này là tập tất cả các chuỗi trong không gian lời giải mà với mọi vị trí trong chuỗi giá trị của gen trùng với giá trị của sơ đồ.

Ví dụ: các chuỗi và sơ đồ có chiều dài 10.

* Sơ đồ (\*111100100) sẽ khớp với hai chuỗi:

{(0111100100), (1111100100)}

Và sơ đồ (\*1\*1100100) sẽ khớp với 4 chuỗi

{(0101100100), (0111100100), (1101100100), (1111100100)}

Đương nhiên, sơ đồ (1001110001) chỉ khớp với chính nó, và sơ đồ (\*\*\*\*\*\*\*\*\*\*) khớp với tất cả các chuỗi có chiều dài 10. Rõ ràng là mỗi sơ đồ cụ thể có tương ứng 2r chuỗi, với r là số ký tự đại diện ‘\*’ có trong sơ đồ. Mặc khác, mỗi chuỗi chiều dài m sẽ khớp với 2m sơ đồ.

Một chuỗi chiều dài m, sẽ có tối đa 2m sơ đồ. Trong một quần thể kích thước n, có thể có tương ứng từ 2 m đến nx2m sơ đồ khác nhau.

Các sơ đồ khác nhau có những đặc trưng khác nhau. Các đặc trưng này thể hiện qua hai thuộc tính quan trọng bậc và chiều dài xác định.

Bậc của sơ đồ S (ký hiệu σ(S)) là chiều dài của chuỗi trừ đi số ký tự đại diện. Bậc xác định đặc trưng của sơ đồ.

Ví dụ: ba sơ đồ chiều dài 10

* S1=(\*\*\*001\*110)

S2=(\*\*\*\*00\*\*0\*)

S3=(11101\*\*001)

Có bậc tương ứng: σ(S1)=6; σ(S2)=3; σ(S3)=8

Khái niệm bậc của sơ đồ giúp cho việc tính xác suất sống còn của sơ đồ do ảnh hưởng của đột biến.

Chiều dài xác định của sơ đồ S (ký hiệu là δ(S)) là khoảng cách giữa hai vị trí cố định ở đầu và cuối. Nó định nghĩa “độ nén” của thông tin chứa trong một sơ đồ.

Ví dụ:

* δ(S1)=10-4=6; δ(S2)=9-5=4; δ(S3)=10-1=9

Như vậy, một sơ đồ chỉ có một vị trí cố định duy nhất thì sẽ có chiều dài xác định là 0.

Khái niệm chiều dài xác định của sơ đồ giúp tính xác suất sống còn của sơ đồ do ảnh hưởng của phép lai.

GAs sử dụng một quần thể của các lời giải có thể. Mỗi lời giải được đại diện bởi một NST, nó chỉ là một đại diện trừu tượng. Các NST được mã hóa thành các chuỗi nhị phân, mỗi vị trí trên chuỗi tồn tại hai giá trị là “1” hoặc “0”. Chẳng hạn như: 1 0 0 1 0 1 0 1 1 0.

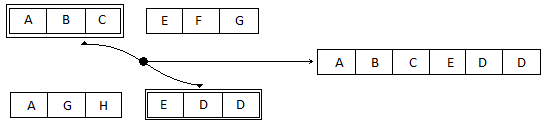
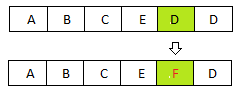
Độ tốt của một cá thể được đánh giá bằng hàm mục tiêu g(x) với x là một NST. Hàm mục tiêu g(x) sau khi được tính toán sẽ là cơ sở để đánh giá độ thích nghi của cá thể. Hàm thích nghi f(x) là sẽ quyết định khả năng một cá thể được chọn lọc vào thế hệ sau, việc ánh xạ g(x)→ f(x) có nhiều phương pháp ánh xạ khác nhau phụ thuộc vào mục đích của bài toán.

1. **Bài toán sinh chuỗi “ To Be Or Not To Be”**

Trạng thái mục tiêu chính là chuỗi “ **To Be Or Not To Be**”

Các trạng thái là các chuỗi được sinh ngẫu nhiên có cùng độ dài. Mỗi chuỗi này biểu diễn cho một chromosome bao gồm các gene nối tiếp nhau, mỗi game là một ký tự. Sự lai tạo có thể được hình dung như sau:

Nếu việc lai tạo được thực hiện ở chính giữa chuỗi, ta có cá thể mới sinh ra bằng cách lấy nữa đầu chuỗi 1 ghép với nữa sau chuỗi 2 để có “ ABCEDD”



Sự biến dị thực chất chỉ là việc thay đổi một số gene trong chuỗi được sinh ra. Ví dụ kí tự đầu của chromosome “ABCE**D**D” bị biến dị thành “ABCE**F**D”

Fitness của một trạng thái có thể định nghĩa là số lượng các ký tự giống với trạng thái mục tiêu

* 1. **Thuật toán**

Khởi tạo quần thể ngẫu nhiên

Lặp:

Tính fitness cho mỗi cá thể so với mục tiêu

Lựa chọn ngẫu nhiên 2 cá thể theo quy luật chọn lọc tự nhiên. Ở đây, sau khi update fitness cho từng cá thể, ta tạo ra một mating pool để lấy 2 phần tử sao ngẫu nhiên từ pool này cho cá thể mạnh hơn có cơ hội được chọn lựa cao hơn.

Ví dụ: Có 5 cá thể A,B,C,D,E lần lượt có fitness như sau: 1,3,2,0,4. Như vậy mating pool như sau: A,B,B,B,C,C,E,E,E,E

Rõ ràng nếu ta lấy ngẫu nhiên 1 phần tử trong pool này tỷ lệ chọn ra E sẽ là cao nhất.Thực hiện lai ghép để tạo cá thể mới.Thay thế cá thể cũ bằng cá thể mới tạo ra

Nếu cá thể mới giống với mục tiêu hoặc giống ở mức chấp nhận được thì dừng. Ở đây rất có thể chúng ta không tìm được kết quả tối ưu hoặc rất lâu mới tìm thấy trạng thái này, như vậy ta có quyền dừng ở trạng thái chấp nhận được để tiết kiệm thời gian.

* 1. **Thiết kế thuật toán**

// khởi tạo quần thể với số lượng n cá thể, mỗi cá thể có chiều dài k

InitializePopulation(int len, int nPopulation);

// khởi tạo quần thể với số lượng n cá thể, mỗi cá thể có chiều dài k

while (foundTarget)

{

// tạo mating pool

var matingPool = UpdateFitnessAndCreateMatingPool();

// lấy 2 phần tử dựa vào mating pool

var p1 = population[matingPool[\_rnd.Next(pool.Count)]];

var p2 = population[matingPool[\_rnd.Next(pool.Count)]];

// ngẫu nhiên tìm điểm cắt

var splitPoint = \_rnd.Next(\_target.Length - 2) + 1;

// lai tạo để tạo ra phần tử mới

var newDNA = p1.CrossOver(p2, splitPoint)

// biến dị dựa trên tỷ lệ biến dị

newDNA.Mutate(mutationRate, materials);

// thay thế phần tử cũ

population[i] = newDNA;

// kiểm tra điều kiện dừng

if (newDNA.CompareTo(target) == target.Length)

foundTarget = true;

}

* Như vậy, ta cần định nghĩa thêm một class DNA để đại diện cho một cá thể trong quần thể:

class DNA

{

Public char[] Content { get; private set; } // chuỗi gene

// lai tạo ra dna mới

Public DNA CrossOver(DNA dna, int splitPoint);

// biến dị theo tỷ lệ

Public void Mutate(double mutationRate);

// hàm tính khoảng cách tới DNA khác

// hỗ trợ việc tính fitness

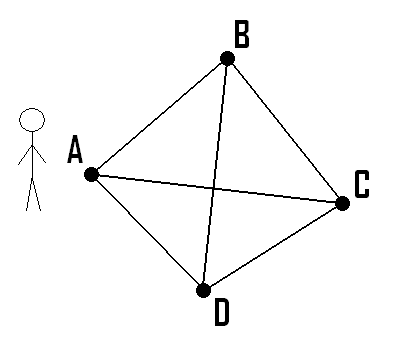
public int CompareTo(char[] content); }

**CHƯƠNG 4: ÁP DỤNG GIẢI THUẬT DI TRUYỀN VÀO BÀI TOÁN NGƯỜI BÁN HÀNG**

1. **Tổng quan bài toán người bán hàng**

Bài toán người bán hàng (tiếng Anh: travelling salesman problem - TSP) là một bài toán thuộc thể loại [tối ưu rời rạc](https://vi.wikipedia.org/w/index.php?title=T%E1%BB%91i_%C6%B0u_r%E1%BB%9Di_r%E1%BA%A1c&action=edit&redlink=1) hay [tổ hợp](https://vi.wikipedia.org/w/index.php?title=T%E1%BB%91i_%C6%B0u_t%E1%BB%95_h%E1%BB%A3p&action=edit&redlink=1) được nghiên cứu trong [vận trù học](https://vi.wikipedia.org/wiki/V%E1%BA%ADn_tr%C3%B9_h%E1%BB%8Dc) hoặc [lý thuyết khoa học máy tính](https://vi.wikipedia.org/w/index.php?title=L%C3%BD_thuy%E1%BA%BFt_khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh&action=edit&redlink=1). Bài toán được phát biểu như sau. Cho trước một danh sách các thành phố và khoảng cách giữa chúng, tìm chu trình ngắn nhất thăm mỗi thành phố đúng một lần.

Bài toán được nêu ra lần đầu tiên năm 1930 và là một trong những bài toán được nghiên cứu sâu nhất trong tối ưu hóa. Nó thường được dùng làm thước đo cho nhiều phương pháp tối ưu hóa. Mặc dù bài toán rất khó giải trong trường hợp tổng quát, có nhiều phương pháp giải chính xác cũng như [heuristic](https://vi.wikipedia.org/wiki/Heuristic) đã được tìm ra để giải quyết một số trường hợp có tới hàng chục nghìn thành phố



Hình 5: Minh họa bài toán người bán hàng

1. **Giải bài toán bán hàng bằng giải thuật di truyền**

Bài toán người bán hàng (traveling salesperson problem – TSP) là một bài toán cổ điển đối với AI và khoa học máy tính

Phát biểu của bài toán TSP: Một người bán hàng có nhiệm vụ ghé thăm N thành phố như là một phần của lộ trình bán hàng. Đường đi giữa mỗi cặp thành phố có một chi phí (ví dụ như độ dài đoạn đường, giá vé máy bay). Hãy tìm ra đường đi có chi phí thấp nhất cho người bán hàng để bắt đầu lên đường tại một thành phố, thăm tất cả các thành phố khác chỉ đúng một lần rồi quay lại thành phố xuất phát.

Toàn bộ không gian trạng thái của bài toán đòi hỏi phải xem xét N! trạng thái để có thể tìm ra lời giải tối ưu, trong đó N là số thành phố cần đi qua. Khi N khá lớn thì bài toán sẽ bị bùng nổ tổ hợp, vì vậy người ta đặt vấn đề là có cần thiết hay không cho việc chạy một máy trạm làm việc đắt tiền trong nhiều giờ để cho một lời giải tối ưu hay chỉ nên chạy một PC rẻ tiền trong vài phút để có được những kết quả “đủ tốt”. Giải thuật di truyền chính là một giải pháp cho lựa chọn thứ hai.

Ở bài toán này, dùng mẫu bit để biểu diễn cho lời giải của bài toán không phải là một cách hay. Chẳng hạn, ta có chín thành phố cần ghé thăm 1, 2, …9, ta xem mỗi thành phố như một mẫu 4 bit 0001, 0010,1001. Khi đó một lời giải khả dĩ sẽ có hình thức như sau:

0001 0010 0011 0100 0101 0110 0111 1000 1001

Với cách biểu diễn như vậy, việc thiết kế các toán tử di truyền sẽ trở nên rất khó khăn. Toán tử lai ghép nhất định là không được, vì chuỗi mới được tạo từ hai cha mẹ khác nhau hầu như sẽ không biểu diễn một đường đi trong đó ghé thăm mỗi thành phố đúng một lần. Trong thực tế, với lai ghép, một số thành phố có thể bị xóa bỏ trong khi các thành phố khác được ghé thăm nhiều hơn một lần, và vì vậy đó không phải là một lời giải hợp lệ. Còn toán tử đột biến thì thế nào? Giả sử bit trái nhất của thành phố thứ sáu, 0110, được đột biến thành 1? 1110, hay là 14, thì nó không còn là một thành phố hợp lệ.

Một cách tiếp cận khác là sẽ bỏ qua biểu diễn dạng mẫu bit và đặt cho mỗi thành phố một tên theo bảng chữ cái hoặc số, ví dụ 1, 2, …9; xem đường đi qua các thành phố là một sự sắp thứ tự của chín ký số này, và sau đó chọn toán tử di truyền thích hợp để tạo ra các đường đi mới. Ở đây ta thấy phép trao đổi (exchange) ngẫu nhiên hai thành phố trong đường đi có thể sử dụng được, còn phép toán lai ghép (crossover) thì không. Việc trao đổi các đoạn của một đường đi với những đoạn khác của cùng đường đi đó, hoặc bất cứ toán tử nào sắp xếp lại các chữ cái của đường đi ấy (mà không xóa bỏ, thêm, hay nhân đôi bất cứ thành phố nào) đều có thể sử dụng được. Tuy nhiên, những phương pháp này gây khó khăn cho việc đưa vào thế hệ con cháu những thành phần “tốt hơn” của các mẫu trong các đường đi qua của các thành phố của hai cha mẹ khác nhau.

Nhiều nhà nghiên cứu đã đưa ra các toán tử lai ghép có khả năng khắc phục những vấn đề này, trong đó có toán tử lai ghép có thứ tự (order crossover) do Davis đưa ra vào năm 1985. Lai ghép có thứ tự xây dựng con cháu bằng cách chọn một dãy con các thành phố trong đường đi của một mẫu cha mẹ. Nó cũng bảo toàn thứ tự tương đối các thành phố từ cha mẹ kia. Đầu tiên, chọn hai điểm cắt, biểu thị bởi dấu “|”, điểm cắt này được chen vào một cách ngẫu nhiên vào cùng một vị trí của mỗi mẫu cha mẹ. Những điểm cắt này là ngẫu nhiên, nhưng một khi được chọn, thì những vị trí như nhau sẽ được sử dụng cho cả hai cha mẹ. Ví dụ, có hai mẫu cho mẹ p1 và p2, với các điểm cắt sau thành phố thứ ba và thứ bảy:

p1 = (1 9 2 | 4 6 5 7 | 8 3)

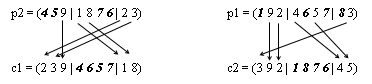
p2 = (4 5 9 | 1 8 7 6 | 2 3)

Hai mẫu con c1 và c2 sẽ được sinh ra theo cách sau. Đầu tiên, các đoạn giữa hai điểm cắt sẽ được chép vào các mẫu con:

c1 = (x x x | 4 6 5 7 | x x)

c2 = (x x x | 1 8 7 6 | x x)

Bước kế tiếp là bắt đầu từ điểm cắt thứ hai của một trong hai mẫu cha mẹ, nếu ta đang muốn hoàn tất mẫu c1, thì ta sẽ bắt đầu từ điểm cắt thứ hai của mẫu p2, ta chép các thành phố từ điểm cắt này theo thứ tự vào các chỗ còn trống của c1, bỏ qua những thành phố mà c1 đã có (các ký số được in đậm và nghiêng trong sơ đồ bên dưới). Khi đến cuối mẫu p2, thì quay lại đầu mẫu p2 tiếp tục chép sang c1 cho đến khi c1 đủ.



Với giải thuật lai ghép này, các đường đi của thế hệ con sẽ được đảm bảo là các đường đi hợp lệ, đi qua mỗi thành phố một lần duy nhất.

Tóm lại, trong lai ghép thứ tự, các mảnh của một đường đi được truyền từ một cha mẹ, p1, sang một con, c1, trong khi sắp xếp của các thành phố còn lại của con c1 được thừa kế từ cha mẹ kia, p2. Điều này ủng hộ cho trực giác hiển nhiên là thứ tự của các thành phố đóng vai trò quan trọng trong việc tạo ra đường đi với chi phí thấp nhất, và vì vậy việc truyền lại các đoạn thông tin có thứ tự này từ các cha mẹ có độ thích nghi cao sang con cái là một điều rất quan trọng.

**CHƯƠNG 5: KẾT LUẬN**

Với ví dụ vừa nêu trên làm nổi bật những vấn đề mang tính duy nhất của thuật toán di truyền về biểu diễn tri thức, chọn toán tử di truyền, và thiết kế hàm thích nghi. Biểu diễn được chọn phải hỗ trợ cho các toán tử di truyền. Một điểm dáng lưu ý nữa là các toán tử di truyền phải được thiết kế sao cho bảo lưu được những mảnh thông tin có ý nghĩa trong lời giải tiềm năng từ thế hệ này sang các thế hệ tiếp theo.

Về cơ bản, nghiên cứu đã đạt được những yêu cầu đã đặt ra. Tuy nhiên vì thời gian cũng như kiến thức có hạn nên đề tài này còn nhiều thiếu sót. Nhóm rất mong nhận được sự góp ý đánh giá từ thầy để đề tài chúng em được hoản thiện hơn. Nhóm em xin chân thành cảm ơn.

**CHƯƠNG 6: TÀI LIỆU THAM KHẢO**

* <https://quangnle.com/genetic-algorithm-thuat-toan-di-truyen>
* <https://nerophung.github.io/2020/05/28/genetic-algothrm>
* <https://tapchikhoahoc.dnpu.edu.vn/UserFiles/Docs/TapChi/2016/1/10>
* <https://vi.wikipedia.org>