TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TT & TT
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN KHOA HỌC MÁY TÍNH
****

BÀI GIẢNG

GIẢI THUẬT DI TRUYỀN & ỨNG DỤNG

Giảng viên biên soạn:

1. **ThS Nguyễn Hiền Trinh**
2. **TS Vũ Vinh Quang**
3. **ThS Nguyễn Thu Huyền**

Thái nguyên, năm 2013

MỤC LỤC

Chương 1
CÁC KHÁI NIỆM CƠ BẢN

1.1. Giới thiệu chung

Giải thuật di truyền GA(GENETIC ALGORITHM**)** do D.E. Goldberg đề xuất, sau đó được L. Davis và Z. Michalevicz phát triển, đây cũng chính là một trong các thuật toán tiến hóa. Thuật toán tiến hóa là các chương trình máy tính có dùng các thuật toán tìm kiếm, tối ưu hóa dựa trên nguyên lý tiến hóa tự nhiên.

Giải thuật di truyền được hình thành dựa trên quan niệm: *quá trình tiến hóa tự nhiên là quá trình hoàn hảo và hợp lý nhất, tự quá trình này đã mang tính tối ưu*. Quan niệm này là một tiên đề đúng, không chứng minh được nhưng phù hợp với thực tế khách quan. Tính tối ưu của quá trình tiến hóa thể hiện ở đặc điểm, thế hệ sau bao giờ cũng tốt hơn (phát triển hơn, hoàn thiện hơn) thế hệ trước. Tiến hóa tự nhiên được duy trì nhờ hai quá trình cơ bản là sinh sản và chọn lọc tự nhiên, trong suốt quá trình tiến hóa tự nhiên, các thế hệ mới luôn được sinh ra để bổ sung thay thế thế hệ cũ. Cá thể nào phát triển hơn, thích ứng hơn với môi trường sẽ tồn tại, cá thể nào không thích ứng được với môi trường sẽ bị đào thải. Sự thay đổi của môi trường là động lực thúc đẩy quá trình tiến hóa, ngược lại tiến hóa cũng tác động trở lại góp phần thay đổi môi trường.

Giải thuật di truyền (GA-Genetic Algorithms) là giải thuật tìm kiếm, chọn lựa các giải pháp tối ưu để giải quyết các bài toán thực tế khác nhau, dựa trên cơ chế chọn lọc của tự nhiên: từ tập lời giải ban đầu, thông qua nhiều bước tiến hoá, hình thành tập lời giải mới phù hợp hơn, và cuối cùng dẫn đến lời giải tối ưu toàn cục.

Trong tự nhiên, mỗi cá thể muốn tồn tại và phát triển phải thích nghi với môi trường, cá thể nào thích nghi hơn thì tồn tại, cá thể nào kém thích nghi thì bị tiêu diệt. Trong mỗi cá thể, các gen liên kết với nhau theo cấu trúc dạng chuỗi, gọi là nhiễm sắc thể (NST). Mỗi NST đặc trưng cho mỗi loài và quyết định sự sống còn của cá thể đó. Do môi trường tự nhiên luôn biến đổi nên cấu trúc NST cũng thay đổi để thích nghi với môi trường và thế hệ sau luôn thích nghi hơn thế hệ trước. Cấu trúc này có được do sự trao đổi thông tin có tính ngẫu nhiên với môi trường bên ngoài hoặc giữa các NST với nhau.

Từ ý tưởng đó, các nhà khoa học đã nghiên cứu và xây dựng nên giải thuật di truyền dựa trên cơ sở chọn lọc tự nhiên và quy luật tiến hoá. Giải thuật di truyền sử dụng các thuật ngữ được lấy từ di truyền học như: lai ghép, đột biến, NST, cá thể...Ở đây mỗi cá thể được đặc trưng bởi một tập nhiễm sắc thể, nhưng để đơn giản khi trình bày, ta xét trường hợp tế bào mỗi cá thể chỉ một NST. Các NST được chia nhỏ thành các gen được sắp xếp theo một dãy tuyến tính. Mỗi cá thể (hay NST) biểu diễn một lời giải có thể của bài toán. Một xử lý tiến hoá duyệt trên tập các NST tương đương với việc tìm kiếm lời giải trong không gian lời giải của bài toán. Quá trình tìm kiếm phải đạt được hai mục tiêu:

Thứ 1: Khai thác những lời giải tốt nhất

Thứ 2: Xem xét trên toàn bộ không gian tìm kiếm

Giải thuật di truyền có thể mô tả vắn tắt bằng thủ tục tổng quát:

Procedure G*iải_thuật_di_truyền*;
Begin
t:=0;
Khởi tạo ngẫu nhiên quần thể P(t);
Đánh giá độ phù hợp từng cá thể trong P(t);
Repeat
t:=t+1;
Chọn các cá thể từ P(t - 1);
Lai tạo các cá thể đã chọn tạo ra P(t) mới;
Đột biến các cá thể trong P(t) theo xác suất p_m;
Đánh giá độ phù hợp các cá thể trong tập P(t);
Until (thoả điều kiện dừng);
End;

Tại lần lặp thứ t, GA xác định một tập hợp các lời giải có thể (các cá thể hay NST) gọi là quần thể $P(t) = \{ x^t_1, x^t_2, ..., x^t_n \}$. Mỗi lời giải x^t_i được đánh giá nhằm xác định độ phù hợp của nó. Sau đó, một tập hợp các lời giải được hình thành nhờ sự lựa chọn các lời giải phù hợp hơn. Một số phần tử của tập hợp này được tái sản xuất thông qua lai ghép và đột biến. Từ đó hình thành quần thể mới P(t+1) với hy vọng chứa các cá thể phù hợp hơn quần thể trước đó.

Toán tử "*lai ghép*" kết hợp các đặc trưng của hai NST cha và mẹ hình thành hai NST con tương ứng chẳng hạn bằng cách hoán vị các đoạn thích hợp của hai NST cha và mẹ. Ví dụ, nếu cặp nhiễm sắc thể cha mẹ được biểu diễn dưới dạng hai véc tơ:

$(a_1, b_1, c_1, d_1, e_1)$ và $(a_2, b_2, c_2, d_2, e_2)$

thì cặp véc tơ con cháu nhận được sau khi lai ghép có thể là:

$(a_1, b_1, c_1, d_2, e_2)$ và $(a_2, b_2, c_2, d_1, e_1)$

Toán tử "*đột biến*" thay đổi một hay một số gen của NST được chọn theo quy tắc thay đổi ngẫu nhiên với xác suất bằng tỷ lệ đột biến.

Như vậy, bản chất GA là một giải thuật lặp, nhằm giải quyết các bài toán tìm kiếm dựa trên cơ chế chọn lọc nhân tạo và sự tiến hoá của các gen. Trong quá trình đó, sự sống còn của cá thể phụ thuộc vào hoạt động của các NST và quá trình chọn lọc (tham gia vào việc tái tạo ra các chuỗi NST mới bằng cách giải mã các chuỗi NST và tạo ra mối liên kết giữa các NST trong các cá thể khác nhau).

GA sử dụng các toán tử: chọn lọc, lai ghép, đột biến trên các NST để tạo ra chuỗi mới. Những toán tử này thực chất là việc sao chép chuỗi, hoán vị các chuỗi con và sinh số ngẫu nhiên.

Cơ chế của GA đơn giản nhưng lại có sức mạnh hơn các giải thuật thông thường khác nhờ có sự đánh giá và chọn lọc sau mỗi bước thực hiện. Do vậy khả năng tiến gần đến lời giải tối ưu của GA sẽ nhanh hơn nhiều so với các giải thuật khác.

Có thể nói GA khác với những giải thuật tối ưu thông thường ở những đặc điểm sau:

1. GA làm việc với tập mã của biến chứ không phải bản thân biến

2. GA thực hiện tìm kiếm trên một quần thể các cá thể chứ không phải trên một điểm nên giảm bớt khả năng kết thúc tại một điểm tối ưu cục bộ mà không thấy tối ưu toàn cục.

3. GA chỉ cần sử dụng thông tin của hàm mục tiêu để phục vụ tìm kiếm chứ không đòi hỏi các thông tin hỗ trợ khác.

4. Các thao tác cơ bản trong giải thuật dựa trên khả năng tích hợp ngẫu nhiên, mang tính xác suất chứ không tiền định.

Cá thể (hay kiểu gen, cấu trúc) trong một quần thể còn được gọi là các chuỗi hay các nhiễm sắc thể (trong giải thuật di truyền chúng ta chỉ xét những cá thể có một nhiễm sắc thể, điều này khác với mỗi tế bào của một cơ thể của một chủng loại cho trước mang một số những nhiễm sắc thể nào đó, như người có 46 nhiễm sắc thể). Các nhiễm sắc thể được hình thành từ các đơn vị - các gen- biểu diễn trong một chuỗi tuyến tính. Mỗi gen kiểm soát một số đặc trưng. Gen với những đặc trưng nhất định có vị trí nhất định trong nhiễm sắc thể, bất cứ đặc trưng nào của mỗi cá thể có thể tự biểu hiện một cách phân biệt và gen có thể nhận một số giá trị khác nhau (các giá trị về tính năng).

Mỗi kiểu (nhóm) gen (được gọi là một nhiễm sắc thể) sẽ biểu diễn lời giải của bài toán đang giải (ý nghĩa của một nhiễm sắc thể cụ thể được người sử dụng xác định trước). Một tiến trình tiến hóa được thực hiện trên một quần thể các nhiễm sắc thể tương ứng với một quá trình tìm kiếm lời giải trong không gian lời giải, quá trình tìm kiếm này cần cân đối giữa 2 mục tiêu là: Khai thác những lời giải tốt nhất và khảo sát không gian tìm kiếm. Thuật giải di truyền (GA) là phương pháp tìm kiếm (độc lập miền) tạo được sự cân đối giữa việc khai thác và khảo sát không gian tìm kiếm.}

Giải thuật di truyền thuộc lớp các giải thuật xác suất, nhưng chúng lại khác các giải thuật ngẫu nhiên vì GA kết hợp các phần tử tìm kiếm trực tiếp và ngẫu nhiên. Khác biệt quan trọng giữa tìm kiếm của GA và các phương pháp tìm kiếm khác là GA duy trì và xử lý một tập các lời giải (một quần thể), tất cả các phương pháp khác chỉ xử lý một điểm trong không gian tìm kiếm, chính bởi vậy, GA mạnh hơn các phương pháp tìm kiếm hiện có rất nhiều.

Cá thể mới sinh ra trong quá trình tiến hóa nhờ sự lai ghép ở thế hệ cha-mẹ, một cá thể mới có thể những tính trạng của cha-mẹ (di truyền), cũng có thể mang những tính trạng hoàn toàn mới (đột biến). Di truyền và đột biến là hai cơ chế có vai trò quan trọng như nhau trong tiến trình tiến hóa, cho dù đột biến xảy ra với xác suất nhỏ hơn nhiều so với hiện tượng di truyền. Các thuật toán tiến hóa tuy có nhiều điểm khác nhau nhưng đều mô phỏng 4 quá trình cơ bản: Lai ghép, đột biến, sinh sản và chọn lọc tự nhiên.

1. Quá trình lai ghép: Là quá trình hình thành nhiễm sắc thể mới trên cơ sở các nhiễm sắc thể cha-mẹ, bằng cách ghép một hay nhiều đoạn gen của hai (hay nhiều) nhiễm sắc thể cha-mẹ với nhau. Phép lai xảy ra với xác suất p_c, quá trình được mô phỏng:

+ Chọn ngẫu nhiên hai (hay nhiều) cá thể bất kỳ trong quần thể. Giả sử các nhiễm sắc thể của cha-mẹ đều có m-gen

+ Tìm điểm lai bằng cách tạo một số ngẫu nhiên trong khoảng từ 1 đến m-1. Điểm lai sẽ chia các chuỗi nhiễm sắc thể cha-mẹ có độ dài m thành hai nhóm chuỗi con với độ dài là m_1 và m_2, hai chuỗi nhiễm sắc thể con mới là $m_{11}+m_{22}$ và $m_{21}+m_{12}$

+ Đưa hai cá thể mới này vào quần thể để tham gia các quá trình tiến hóa tiếp theo.

2. Quá trình đột biến: Là hiện tượng cá thể con mang một số tính trạng không có trong mã di truyền của cha-mẹ. Phép đột biến xảy ra với xác suất p_m nhỏ hơn rất nhiều so với xác suất lai p_c. Phép đột biến được mô phỏng:

+ Chọn ngẫu nhiên một cá thể bất ỳ cha-mẹ trong quần thể

+ Tạo ngẫu nhiên một số k trong khoảng từ 1 đến m ($1 \le k \le m$)

+ Thay đổi gen thứ k và trả cá thể này về quần thể để tham gia quá trình tiến hóa tiếp theo.

3. Quá trình sinh sản và chọn lọc (Phép tái sinh và phép chọn)

3.1. Phép tái sinh: là quá trình trong đó các cá thể được sao chép trên cơ sở độ thích nghi của nó. Độ thích nghi là một hàm gán một giá trị thực cho các cá thể trong quần thể, quá trình này được mô phỏng:

+ Tính độ thích nghi của từng cá thể trong quần thể, lập bảng cộng dồn các giá trị thích nghi (theo số thứ tự gán cho từng cá thể) ta được tổng độ thích nghi. Giả sử quần thể có n cá thể, gọi độ thích nghi của cá thể thứ i là F_i, tổng dồn thứ i là F_{ti}, tổng độ thích nghi của toàn quần thể là F_m

+ Tạo một số ngẫu nhiên F trong đoạn từ 0 đến F_m

+ Chọn cá thể thứ k đầu tiên thỏa mãn $F \ge F_{tk}$ đưa vào quần thể của thế hệ mới.

3.2 Phép chọn: Là quá rình loại bỏ các cá thể xấu trong quần thể để chỉ giữ lại trong quần thể các cá thể tốt, phép chọn được mô phỏng:

+ Sắp xếp quần thể theo thứ tự độ thích nghi giảm dần

+ Loại bỏ các cá thể ở cuối dãy để chỉ giữ lại n cá thể tốt nhất. Ta giả sử quần thể có kích thước cố định là n.

Giải thuật di truyền giải một bài toán cần có các thành phần sau:

1. Một cấu trúc dữ liệu I biểu diễn không gian lời giải của bài toán
2. Cách khởi tạo quần thể ban đầu
3. Hàm định nghĩa độ thích nghi eval(), đóng vai trò môi trường
4. Các phép toán di truyền (phép lai, phép đột biến, phép tái sinh và phép chọn)
5. Các tham số được giải thuật di truyền sử dụng (kích thước quần thể, xác suất lai, đột biến...)

1.2 Các vấn đề cơ bản của giải thuật di truyền

1.2.1. *Mã hoá - mô tả di truyền cho lời giải của bài toán*

Việc mô tả di truyền cho lời giải cho bài toán gồm hai phần cơ bản:

1. Xây dựng cấu trúc gen cho mỗi lời giải của bài toán để từ mỗi lời giải ta có thể mã hoá thành một NST (chuỗi các gen).

2. Giải mã các NST để nhận được lời giải.

Đây là vấn đề cần giải quyết trước khi giải bài toán với GA. Tuỳ thuộc vào nội dung của mỗi bài toán mà ta có cách mã hoá khác nhau.

Sau đây là các phương pháp mã hoá hay được sử dụng:

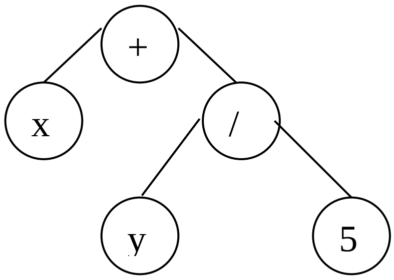
Mã hoá dạng chuỗi nhị phân: đây là phương pháp thông dụng và cơ bản nhất được sử dụng ngay từ bước ban đầu khi nghiên cứu GA. Trong phương pháp này mỗi NST là một chuỗi các bit 0 và 1.

Mã hoá thứ tự: được sử dụng trong bài toán có sắp xếp thứ tự. Ở đây mỗi NST là một chuỗi các số nguyên thể hiện thứ tự phân bố lời giải của bài toán.

Mã hoá theo giá trị: được sử dụng trong các bài toán mà mỗi lời giải là tập các giá trị (ví dụ tập số thực). Trong phương pháp này, mỗi NST là một chuỗi các giá trị có mối quan hệ tương ứng với bài toán.

Mã hoá dạng cây: được sử dụng chủ yếu trong các biểu thức toán học, trong phương pháp mã hoá này mỗi NST là một cây của một nhóm đối tượng nào đó.

Ví dụ: Biểu thức sau x+(y / 5) được mã hoá thành:



Mã hoá số thực : Mỗi NST được mã hoá là một véc tơ trong không gian R^m chẳng hạn $X = (a_1, a_2, ..., a_m)$ với các $a_i \in R$. Cách mã hoá này thường tự nhiên đối với các bài toán tối ưu số và được phát triển rất mạnh trong thời gian gần đây.

1.2.2. *Tạo lập lời giải ban đầu (khởi tạo quần thể)*

Tập lời giải ban đầu thường được khởi tạo ngẫu nhiên từ miền xác định của các lời giải. Cách tạo lập tập lời giải ban đầu phụ thuộc rất nhiều vào cách mã hoá NST.

Với phương pháp mã hoá nhị phân: xây dựng NST bằng cách tạo ngẫu nhiên chuỗi các bit 0 hoặc 1.

Với phương pháp mã hoá thứ tự: xây dựng NST ban đầu bằng cách hoán vị ngẫu nhiên các thứ tự.

Với phương pháp mã hoá theo giá trị: tạo ngẫu nhiên từng giá trị trong miền xác định của lời giải để tạo ra chuỗi NST ban đầu.

Với mã hoá số thực: tạo ngẫu nhiên N véc tơ thực trong R^m.

1.2.3. *Xây dựng hàm phù hợp(hàn thích nghi)*

Hàm phù hợp đánh giá khả năng phù hợp của tập lời giải theo yêu cầu bài toán. Hàm này được xây dựng cho từng bài toán với yêu cầu cụ thể. Thông thường trong các bài toán tối ưu hàm này chính là hàm mục tiêu của bài toán.

1.2.4. *Các toán tử di truyền*

a. *Toán tử chọn lọc*

Trong quá trình thực hiện của giải thuật di truyền, sau mỗi lần tiến hoá ta chỉ giữ lại các cá thể có độ phù hợp cao còn các cá thể phù hợp thấp bị loại bỏ. Toán tử chọn lọc thường giữ lại 50% các cá thể phù hợp nhất. Tuy nhiên người ta cũng phát triển nhiều sơ đồ chọn khác nhau nhằm là tăng tính đa dạng của quần thể, tránh sự hội tụ sớm.

b. *Toán tử lai ghép*

Bước 1: Tạo ra tập NST để tạo sinh từ quần thể bằng cách chọn ngẫu nhiên N NST từ M NST (M là kích cỡ quần thể).

Có nhiều cách chọn:

Chọn ngẫu nhiên theo thứ tự: lặp N lần việc tạo ngẫu nhiên ra một số nguyên i thuộc khoảng [1, M] để chọn NST thứ i.

Chọn theo trọng số: tạo trọng số tích luỹ cho M NST theo công thức:

$$p_i = \frac{i}{\sum_{k=1}^{M} k}$$ (với bài toán tìm min)

$$p_i = \frac{M - i + 1}{\sum_{k=1}^{M} k}$$ (với bài toán tìm max)

Sau khi có trọng số tích luỹ cho NST, ta lần lượt tạo các xác suất ngẫu nhiên r và duyệt từ NST đầu tiên đến khi gặp NST có trọng số tích luỹ lớn hơn r thì chọn nó.

Bước 2: Sau khi chọn được N NST, lần lượt lấy ra từng cặp NST để lai ghép tạo ra hai NST mới. Một số dạng toán tử lai ghép hay dùng là :

Lai ghép 1 điểm: chọn ngẫu nhiên một vị trí sau đó hoán vị phần đứng sau vị trí vừa chọn giữa hai NST cha và mẹ để nhận được hai NST con.

Lai ghép hai điểm: chọn ngẫu nhiên hai vị trí trong một NST, sau đó hoán vị các giá trị đứng giữa hai điểm đã chọn của hai NST cha mẹ để nhận được hai NST con.

Lai ghép mặt nạ: tạo một mặt nạ ngẫu nhiên có số bit bằng chiều dài của NST. Ta sẽ hoán vị các giá trị của hai NST cha và mẹ ở những vị trí tương ứng với vị trí bit 1 của mặt nạ.

c. *Toán tử đột biến*: Toán tử đột biến được xây dựng để tránh việc nhận được giá trị tối ưu cục bộ. Đột biến gây ra thay đổi ngẫu nhiên trên từng bit của NST để tạo ra một NST mới.

d. *Tạo sinh:* Chọn các cá thể từ quần thể hiện thời làm quần thể mới cho lần lặp kế tiếp.

1.3. Các bài toán thực tế

1.3.1 Bài toán tối ưu hàm một biến

Xét bài toán tối ưu không ràng buộc: Max $f(x)=x*\sin(10\pi*x)+1.0$ với $x\in[-1,2]$
Yêu cầu của bài toán là tìm x trong khoảng [-1,2] để f có giá trị lớn nhất, nghĩa là tìm x_0 sao cho $f(x_0)\geq f(x)$, $\forall x\in[-1,2]$
Ta có thể vẽ đồ thị hàm số $f(x)=x*\sin(10\pi*x)+1.0$ và xác định nghiệm theo yêu cầu
Giải:Khi đạo hàm bậc nhất bằng 0, nghĩa là

$$f'(x)=\sin(10\pi*x)+10\pi x*\cos(10\pi*x)=0 \Leftrightarrow \tan g(10\pi*x)=-10\pi x$$

Phương trình trên có vô số lời giải

$$x_i=\frac{2i-1}{20}+\varepsilon_i, i=1,2,...$$

$x_0=0$

$$x_i=\frac{2i+1}{20}-\varepsilon_i, i=-1,-2,...$$

Các số hạng ε_i là các dãy số thực giảm (i=1,2,... và i=-1,-2,...) dần về 0. Ta có hàm f đạt đến cực đại (cục bộ) tại điểm x_i khi i là số nguyên lẻ và đạt đến cực tiểu của nó tại x_i khi i là số nguyên chẵn. Vì miền giá trị của bài toán là [-1,2], hàm đạt cực đại tại $x_{19}=\frac{37}{20}+\varepsilon_{19}=1.85+\varepsilon_{19}$. Lưu ý, $f(x_{19})$ lớn hơn $f(1.85)= 1.85*\sin(18\pi*\pi/2)+1.0=2.85$

Ngoài ra ta sẽ xét cách giải bài toán trên bằng giải thuật di truyền với việc đề cập tới 5 thành phần chính của giải thuật:

1. Biểu diễn: Dùng một vecto nhị phân làm nhiễm sắc thể để biểu diễn các giá trị thực của biến x. Chiều dài của vecto phụ thuộc vào độ chính xác cần có, trong bài này ta tính chính xác đến 6 số lẻ. Miền giá trị của x có chiều dài 2-(-1)=3, với yêu cầu về độ chính xác ta phải chia khoảng [-1,2) thành ít nhất 3 x 10^6 khoảng có kích thước bằng nhau, điều này có nghĩa là cần 22 bít cho vecto nhị phân (nhiễm sắc thể)

$2097152=2^{21}< 3000000 \leq 2^{22}=4194304$

Ánh xạ biến chuỗi nhị phân ($b_{21}b_{20}...b_0$) thành số thực x trong khoảng [-1,2] được thực hiện qua 2 bước:

Bước 1: Đổi chuỗi nhị phân ($b_{21}b_{20}...b_0$) từ cơ số 2 sang cơ số 10:

$$(\langle b_{21}b_{20}...b_0\rangle)_2=\left(\sum_{i=0}^{21} b_i 2^i\right)_{10} = x'$$

Bước 2: Tìm số thực x tương ứng $x=-1+x'.\frac{3}{2^{22}-1}$

Với -1 là cận dưới của miền giá trị và 3 là chiều dài của miền.

Ví dụ: Nhiễm sắc thể (1000101110110101000111) biểu diễn số 0.637197 vì

$x'=(1000101110110101000111)_2=2288967_{10}$ và x=-1.0+2288967

x 3/4194303=0.637197

Lưu ý:

Các nhiễm sắc thể (0000000000000000000000) và (1111111111111111111111) biểu diễn các cận -1 và 2 của miền.

2. Khởi tạo quần thể

Ta tạo một quần thể các nhiễm sắc thể, trong đó mỗi nhiễm sắc thể là một vecto nhị phân 22 bit, tất cả 22 bit của mỗi nhiễm sắc thể đều được khởi tạo ngẫu nhiên.

3. Hàm lượng giá: Hàm lượng giá eval của các vecto nhị phân v chính là hàm f: eval(v)=f(x), trong đó nhiễm sắc thể v biểu diễn giá trị thực x, hàm lượng giá đóng vai trò môi trường, đánh giá từng lời giải theo độ thích nghi của chúng. Xét 3 nhiễm sắc thể:

v_1=(1000101110110101000111)

v_2=(0000001110000000010000)

v_3=(1110000000111111000101) tương ứng với các giá trị x_1=0.637197,

x_2=-0.958973, x_3=1.627888 và có độ thích nghi tương ứng:

eval(v_1)=f(x_1)=1.586345

eval(v_2)=f(x_2)=0.078878

eval(v_3)=f(x_3)=2.250650

Với kết quả nay, ta thấy nhiễm sắc thể v_3 là tốt nhất trong 3 nhiễm sắc thể vì hàm lượng giá của nó trả về giá trị cao nhất.

4. Các phép toán di truyền

Trong giai đoạn tiến hóa quần thể, ta dùng 2 phép toán di truyền cổ điển là: đột biến và lai.

Đột biến làm thay đổi một (số) gen (các vị trí trong nhiễm sắc thể) với xác suất bằng tốc độ đột biến. Giả thiết rằng gen thứ 5 trong nhiễm sắc thể v_3 được chọn để đột biến, đột biến ở đây là thay đổi giá trị của gen này: 0 thành 1 và 1 thành 0. Như vậy, sau đột biến này, v_3 sẽ là v'_3=(1110**1**00000111111000101). Nhiễm sắc thể này biểu diễn giá trị x'_3=1.721638 và $f(x')$=-0.082257. Như vậy, đột biến cụ thể này làm giảm khá nhiều giá trị của nhiễm sắc thể v_3. Ta xét tiếp, gen thứ 10 của v_3 được chọn để đột biến thì v''_3=(1110000000111111000101), nhiễm sắc thể này biểu diễn giá trị x''_3=1.630818 và $f(x'')$=2.343555 lớn hơn giá trị $f(x_3)$=2.250650.

Ta xét phép lai trên các nhiễm sắc thể v_2 và v_3, giả thiết điểm lai được chọn (ngẫu nhiên) ở vị trí thứ 6.

v_2=(00000 | 01110000000010000)

v_3=(11100 | 00000111111000101) , hai con của kết quả lai là

v'_2=(00000 | 00000111111000101))

v'_3=(11100 | 01110000000010000) , các con này có độ thích nghi

$f(v'_2)=f(-0.998113)=0.940865$

$f(v'_3)=f(1.666028)=2.459245$, như vậy con lai thứ 2 v'_3 là thích nghi hơn cả cha và mẹ của nó.

5. Các tham số: Trong bài toán này ta đã dùng các tham số sau

+ Kích thước quần thể pop-size=50

+ Xác suất lai p_c=0.25 . Điều này có nghĩa là cá thể v trong quần thể có 25% cơ hội được chọn để thực hiện phép lai.

+ Xác suất đột biến p_m=0.01, có nghĩa là 1%=1bít bất kỳ của một cá thể bất kỳ trong quần thể bị đột biến.

6. Các kết quả thử nghiệm

Sau khi dùng giải thuật di truyền với 150 thế hệ ta có bảng kết quả:

Thế hệ thứ	Hàm lượng giá
1	1.441942
6	2.250003
8	2.250283
9	2.250284
10	2.250363

12	2.328077
39	2.344251
40	2.345087
51	2.738930
99	2.849246
137	2.850217
145	2.850227

Theo bảng kết quả trên, nhiễm sắc thể tốt nhất sau 150 thế hệ là v_{max}=(1110011010001000000101), tương ứng với giá trị x_{max}=1.850773.

Như vậy, x_{max}=1.85+ε và f(x_{max})> 2.85 một chút.

1.3.2 Bài toán thế tiến thoái lưỡng nam của tù nhân

Xét trò chơi Tiến thoái lưỡng nam của tù nhân, bài toán được phát biểu như sau: Hai tù nhân bị giam trong hai xà lim riêng biệt, không thể thông tin cho nhau. Mỗi tù nhân được yêu cầu là hãy ly khai và phản bội người tù kia, một cách độc lập. Nếu chỉ có một người ly khai, anh ta được thưởng, còn người kia bị phạt. Nếu cả hai đều ly khai, cả hai đều bị giam lại và tra tấn. Nếu không ai ly khai, cả hai đều được phần thưởng lớn. Như vậy, cách chọn ích kỷ là ly khai sẽ luôn đem lại phần thưởng hoặc hình phạt cao hơn, bất chấp người kia chọn ra sao, nhưng nếu cả hai đều ly khai thì họ đều làm những điều tệ hại hơn là hợp tác với nhau. Tình thế khó khăn của người tù là phải quyết định nên ly khai hay hợp tác với người kia.

Trò chơi này gồm 2 người chơi, trong trò chơi, khi tới phiên mình, mỗi người sẽ chọn ly khai hoặc hợp tác với người kia. Các đấu thủ sẽ được ghi điểm theo cách thưởng phạt được liệt kê trong bảng:

Đấu thủ 1	Đấu thủ 2	P_1	P_2	Lời bình
Ly khai	Ly khai	1	1	Phạt vì ly khai lẫn nhau
Ly khai	Hợp tác	5	0	Cám dỗ ly khai và sự thưởng phạt
Hợp tác	Ly khai	0	5	Sự thưởng phạt và cám dỗ ly khai
Hợp tác	Hợp tác	3	3	Phần thưởng cho sự hợp tác

Tiếp cận bài toán bằng giải thuật di truyền nhằm duy trì quần thể các đấu thủ, mỗi cá thể có chiến lược riêng của nó. Ban đầu, chiến lược của mỗi đấu thủ được chọn ngẫu nhiên, sau đó ở mỗi bước mỗi đấu thủ chơi và được ghi điểm. Rồi có một số đấu thủ được chọn cho thế hệ mới và một trong số đó được chọn để ghép đôi, đấu thủ mới được tạo ra có chiến lược thừa hưởng từ những chiến lược của cha-mẹ (lai tạo). Một đột biến, như thường lệ, dẫn tới có biến đổi trong chiến lược của các đối thủ do các thay đổi ngẫu nhiên trên biểu diễn của các chiến lược này.

1. Biễu diễn chiến lược

Ta cần có một số cách biểu diễn về chiến lược (một lời giải khả thi). Để đơn giản, ta sẽ quan tâm đến các chiến lược tất định và sử dụng kết quả của chuyển dịch của 3 bước trước để quyết định bước hiện tại. Vì có 4 khả năng có thể có cho mỗi bước, nên có tất cả 4 x 4 x 4= 64 khả năng chuyển dịch của 3 bước trước đó.

Có thể đặc tả chiến lược theo cách chỉ định chuyển dịch nào cần được thực hiện cho các khả năng có thể xảy ra. Như thế, một chiến lược có thể được biểu diễn bằng chuỗi 64 bit (mỗi bít nhận giá trị Ds-ly khai hoặc Cs-hợp tác), chỉ định cho chuyển dịch nào phải làm đối với 64 khả năng. Để có chiến lược khởi điểm ở đầu một chuyển dịch, chúng ta cần đặc tả tiền đề về 3 chuyển dịch giả định trước khi bắt đầu trò chơi, điều này đòi hỏi phải có hơn 6 gen, như vậy toàn bộ nhiễm sắc thể ít nhất cần 70 bít. Chuỗi 70 bít đặc tả những gì một đấu thủ trong từng trường hợp có thể xảy ra và như thế xác định hoàn toàn một chiến lược riêng. Chuỗi 70 gen chính là một nhiễm sắc thể của đấu thủ để dùng trong tiến trình tiến hóa.

2. Giải thuật di truyền

Giải thuật di truyền dùng để học chiến lược tiến thoái lưỡng nam của tù nhân gồm 4 giai đoạn:

2.1 Chọn quần thể khởi đầu: Mỗi đấu thủ được gán ngẫu nhiên một chuỗi 70 bít, biễu diễn cho một chiến lược của đấu thủ.

2.2 Kiểm tra mỗi đấu thủ để quyết định hiệu quả đạt được: Mỗi đấu thủ dùng một chiến lược được xác định trong nhiễm sắc thể của nó để đấu với những đối thủ khác. Điểm của đấu thủ là điểm trung bình của tất cả các bước.

2.3 Chọn các đấu thủ để phát sinh: Nếu đấu thủ có điểm trung bình sẽ được cho một điểm cộng. Khi điểm có độ lệch chuẩn trên trung bình sẽ được cho hai điểm cộng, còn khi điểm có độ lệch chuẩn dưới trung bình sẽ không được điểm cộng nào.

2.4 Những đấu thủ chơi thành công sẽ được kết đôi một cách ngẫu nhiên để sinh con qua mỗi lần ghép đôi. Chiến lược của mỗi con sẽ được quyết định từ những chiến lược của cha-mẹ. Điều này được thực hiện bằng cách sử dụng hai phép toán di truyền: lai và đột biến.

Sau 4 giai đoạn này ta có một quần thể mới. Quần thể này sẽ biểu diễn các mẫu ứng xử giống cách cư xử của những cá thể thành công trong thế hệ trước nhiều hơn, ít giống cách cư xử của những cá thể không thành công hơn. Với mỗi thế hệ mới, những cá thể có điểm tương đối cao sẽ có nhiều khả năng truyền các phần chiến lược của chúng, trong khi những cá thể không thành công sẽ không có phần chiến lược nào được truyền lại cho thế hệ sau.

3. Kết quả thực nghiệm: Từ một khởi đầu rất ngẫu nhiên, giải thuật di truyền làm tiến hóa các quần thể mà thành viên trung gian cũng thành công, giải thuật dùng các heuristic:

+ Đừng lắc lư thuyền: Tiếp tục hợp tác sau 3 lần tương tác (nghĩa là C, sau (CC)(CC)(CC))

+ Khiêu khích: Ly khai khi đấu thủ khác ly khai (nghĩa là D, sau khi nhận (CC)(CC)(CD))

+ Nhận lời xin lỗi: Tiếp tục hợp tác sau khi việc hợp tác được phục hồi (nghĩa là C, sau (CD)(DC)(CC))

+ Quên: Hợp tác khi sự tương tác đã được phục hồi sau khi bị lợi dụng (nghĩa là C, sau (DC)(CC)(CC))

+ Nhận ghép đôi: Ly khai sau 3 lần ly khai lẫn nhau (nghĩa là D, sau (DD)(DD)(DD))

1.3.3 Bài toán người du lịch (TSP)

Một người du lịch muốn thăm mọi thành phố anh ta quan tâm, mỗi thành phố thăm qua đúng một lần, rồi trở về điểm khởi hành. Biết trước chi phí di chuyển giữa hai thành phố bất kỳ. Hãy xây dựng một lộ trình thỏa các điều kiện trên với tổng chi phí nhỏ nhất.

TSP là bài toán tối ưu tổ hợp và có rất nhiều ứng dụng. Có thể giải bài toán này bằng nhiều phương pháp: Phương pháp nhánh cận, phương pháp gần đúng hay những phương pháp tìm kiếm heuristic. Sau đây ta sẽ tiếp cận giải bài toán theo giải thuật di truyền.

1. Ta đánh số các thành phố và dùng một vecto các thành phần nguyên v để biểu diễn một nhiễm sắc thể lộ trình, $v=(i_1, i_2, ..., v_n)$ biểu diễn một lộ trình: đi từ thành phố i_1 đến thành phố i_2, ..., từ i_{n-1} đến i_n và trở về i_1 (v là một hoán vị của vecto (1, 2, ..., n).

2. Khởi tạo quần thể: Ta có thể sử dụng một số giải thuật heuristic (như dùng giải thuật Greedy áp dụng nguyên lý "tham lam" vào bài toán TSP). Áp dụng giải thuật Greedy nhiều lần, mỗi lần từ một thành phố đầu khác nhau, hoặc đơn giản hơn là khởi tạo quần thể bằng cách tạo ngẫu nhiên các mẫu từ các hoán vị của (1, 2, ..., n).

3. Việc lượng giá một nhiễm sắc thể: Bài toán đã cho trước chi phí của chuyến đi giữa các thành phố, ta có thể dễ dàng tính tổng chi phí của toàn bộ lộ trình.

4. Các phép toán di truyền:

+ Phép lai OX: Cho trước hai cá thể cha-mẹ, cá thể con có được bằng cách chọn thứ tự lộ trình từ một cá thể và bảo toàn thứ tự tương đối giữa các thành phố trong cá thể kia.

Ví dụ 1: Xét hai cha-mẹ (với hai điểm cắt được đánh dấu)

p_1= (1 2 3 |4 5 6 7 | 8 9) và p_2 = (4 5 2 |1 8 7 6 | 9 3)

tạo ra các con như sau:

Đầu tiên các đoạn giữa các điểm cắt được sao chép vào các con:

o_1= (x x x |4 5 6 7 | x x) và o_2=(x x x |1 8 7 6 | x x)

Tiếp theo, bắt đầu từ điểm cắt thứ hai của một cha-mẹ, các thành phố từ cha mẹ kia được sao chép theo cùng thứ tự, bỏ đi các dấu hiệu đã có. Đến cuối chuỗi, ta tiếp tục từ vị trí đầu tiên của chuỗi. Thứ tự của các thành phố trong cha-mẹ thứ hai (từ điểm cắt thứ hai là: 9-3-4-5-2-1-8-7-6, sau khi bỏ đi các thành phố 4, 5, 6, 7 đã có trong con thứ nhất, ta có: 9-3-2-1-8. Thứ tự này được đặt vào con thứ nhất (bắt đầu từ điểm cắt từ thứ hai): o_1= (2 1 8 |4 5 6 7 | 9 3). Tương tự ta có:

Thứ tự 8-9-1-2-3-4-5-6-7, bỏ đi các thành phố 1, 6, 7, 8 đã có trong con thứ hai, ta có 9-2-3-4-5. Thứ tự này được đặt vào con thứ hai bắt đầu từ điểm cắt thứ hai:

o_2=(3 4 5 |1 8 7 6 | 9 2)

Như vậy, lai tạo OX khai thác thuộc tính về biểu diễn đường dẫn mà thứ tự của các thành phố (không phải là vị trí của chúng) rất quan trọng, nghĩa là hai hành trình:

9-3-4-5-2-1-8-7-6 và 4-5-2-1-8-7-6-9-3 thực ra là giống nhau.

Ví dụ 2: Xét hai cha-mẹ (với hai điểm cắt được đánh dấu)

p_1= (1 2 3 |4 5 6 7 | 8 9 10 11 12) và p_2 = (7 3 1 |11 4 12 5| 2 10 9 6 8)

tạo ra các con như sau:

Đầu tiên các đoạn giữa các điểm cắt được sao chép vào các con:

o_1= (x x x |4 5 6 7 | x x x x x) và o_2=(x x x |11 4 12 5 | x x x x x)

Thứ tự của các thành phố trong cha-mẹ thứ hai (từ điểm cắt thứ hai là: 2-10-9-6-8-7-3-1-11-4-12-5, sau khi bỏ đi các thành phố 4, 5, 6, 7 đã có trong con thứ nhất, ta có: 2-10-9-8-3-1-11-12. Thứ tự này được đặt vào con thứ nhất (bắt đầu từ điểm cắt từ thứ hai): o_1= (1 11 12 |4 5 6 7 | 2 10 9 8 3). Tương tự ta có:

Thứ tự 8-9-10-11-12-1-2-3-4-5-6-7, bỏ đi các thành phố 4, 5,11,12 đã có trong con thứ hai, ta có 8-9-10-1-2-3-6-7. Thứ tự này được đặt vào con thứ hai bắt đầu từ điểm cắt thứ hai:

o_2=(3 6 7 |1 8 7 6 | 8 9 10 1 2)

+ Phép đột biến được thực hiện bằng cách hoán vị hai vị trí bất kỳ trong cá thể được chọn.

Và như vậy, với các tham số liên quan được cho trước thì giải thuật di truyền theo phân tích trên sẽ được thực hiện.

1.4. Giải thuật leo đồi, mô phỏng việc luyện thép và di truyền

Phương pháp leo đồi dùng kỹ thuật lặp và áp dụng cho một điểm duy nhất (điểm hiện hành trong không gian tìm kiếm). Trong mỗi bước lặp, một điểm mới được chọn từ lân cận của điểm hiện hành (leo đồi còn được gọi là phương pháp tìm kiếm lận cận hay tìm kiếm cục bộ). Nếu điểm mới cho giá trị (của hàm mục tiêu f) tốt hơn, điểm mới sẽ trở thành điểm hiện hành. Nếu không, một lân cận khác sẽ được chọn và thử. Quá trình trên sẽ dừng nếu không cải thiện thêm được cho lời giải hiện hành. Như vậy, phương pháp leo đồi chỉ cung cấp các giá trị tối ưu cục bộ và các giá trị này phụ thuộc rất nhiều vào điểm khởi đầu. Hơn nữa, không có thêm giả thiết gì về sai số tương đối (thỏa tối ưu toàn cục) của lời giải tìm được. Để tăng cơ hội thành công, phương pháp này được thực hiện nhiều lần, mỗi lần với một điểm khởi đầu khác nhau (các điểm này không cần phải chọn ngẫu nhiên, có một tập hợp các điểm khởi đầu của một lần thực thi phụ thuộc vào kết quả của những lần chạy trước đó).

Kỹ thuật mô phỏng luyện thép là một kỹ thuật khắc phục những nhược điểm của phương pháp leo đồi. Lời giải không còn phụ thuộc nhiều vào điểm khởi đầu và thường là gần với điểm tối ưu. Đạt được điều này là nhờ việc đưa vào xác suất nhận p, xác suất p là hàm theo giá trị của hàm mục tiêu đối với điểm hiện hành và điểm mới và một tham số điều khiển bổ sung, tham số "nhiệt độ" T. Nói chung, nhiệt độ T càng thấp thì cơ hội nhận điểm mới càng nhỏ. Khi thực hiện giải thuật, nhiệt độ T của hệ thống sẽ được hạ thấp dần theo từng bước. Giải thuật dừng khi T nhỏ hơn một ngưỡng cho trước, với ngưỡng này thì gần như không còn thay đổi nào được chấp nhận nữa.

Giải thuật di truyền thực hiện quá trình tìm kiếm lời giải tối ưu theo nhiều hướng, bằng cách duy trì một quần thể các lời giải và thúc đẩy sự hình thành, trao đổi thông tin giữa các hướng này. Quần thể trải qua tiến trình tiến hóa ở mỗi thế hệ lại tái sinh các lời giải tương đối tốt, trong khi các lời giải tương đối xấu thì chết đi. Để phân biệt các lời giải khác nhau, hàm mục tiêu được dùng để đóng vai trò môi trường.

Bài toán: Hàm mục tiêu f cần cực đại hóa $f(v)=| 11* one(v) -150 |$

Với v là các chuỗi nhị phân có chiều dài 30, hàm one(v) cho biết tổng số số 1 có trong chuỗi v. Ví dụ, với 3 chuỗi sau

v_1=(110110101110101111111011011011)

v_2=(111000100100110111001010100011)

v_3=(000010000011001000000010001000)

Thì

$f(v_1)=)=| 11* 22 -150 | =192$

$f(v_2)=)=| 11* 15 -150 | =15$

$f(v_3)=)=| 11* 6 -150 | =84$

Ta thấy f là hàm tuyến tính, hàm f có một cực đại toàn cục với

v_g=(111111111111111111111111111111) khi đó $f(v_g)=)=| 11* 30 -150 | =180$

Và một cực đại cục bộ khi

v_l=(000000000000000000000000000000) khi đó $f(v_g)$=)=| 11* 0 -150 | =150

Dùng bài toán trên để minh họa cho các giải thuật: Leo đồi, luyện thép và di truyền.

1.4.1 Giải thuật leo đồi

Có nhiều phiên bản của giải thuật leo đồi, chúng chỉ khác nhau theo cách chuỗi mới được chọn để so sánh với chuỗi hiện tại. Ta xét thủ tục sau:

Procedure Leodoi;

Begin

1. T:=0;

2. **Lặp lại**

+ Cục bộ:= False;

+ Chọn ngẫu nhiên một chuỗi v_c

+ Tiến hóa v_c

+ *Lặp lại*

+ Chọn 30 chuỗi mới trong lân cận v_c bằng cách thay một trong các bít của v_c

+ Chọn chuỗi v_n trong tập các chuỗi có giá trị hàm mục tiêu f lớn nhất

+ Nếu $f(v_c)<f(v_n)$ thì $v_c:=v_n$

Ngược lại thì Cục bộ:=True;

Cho đến khi (Cục bộ)

+ t:=t+1;

Cho đến khi t=max;

End;

Thủ tục trên được thực hiện max lần lặp. Ban đầu 30 lân cận được quan tâm và lân cận v_n trả về giá trị lớn nhất $f(v_n)$ được chọn để cạnh tranh với chuỗi v_c hiện hành. Nếu $f(v_c)<f(v_n)$ thì chuỗi mới trở thành chuỗi hiện hành. Ngược lại thì không có cải thiện cục bộ nào xảy ra. Giải thuật đạt đến (cục bộ hoặc toàn cục) tối ưu (cục bộ=True), trong trường hợp như vậy thì lần lặp kế tiếp(t=t+1) được thực hiện với một chuỗi mới, chọn theo ngẫu nhiên

Việc thành công hay thất bại của giải thuật leo đồi (nghĩa là trả về tối ưu cục bộ hay toàn cục) tùy thuộc chuỗi khởi đầu (được lựa chọn ngẫu nhiên). Rõ ràng là nếu chuỗi khởi đầu có 13 số 1 hoặc ít hơn, giải thuật sẽ luôn luôn chấm dứt tại tối ưu cục bộ (thất bại). Lý do là chuỗi 13 số 1 trả về giá trị 7 của hàm mục tiêu và bất cứ cải thiện của từng bước lặp đều hướng về tối ưu cục bộ, nghĩa là tăng số 1 đến 14 và giảm trị số của hàm mục tiêu về 4. Mặt khác, bất cứ việc giảm lượng số 1 nào cũng sẽ làm tăng trị số của hàm: Một chuỗi 12 số 1 sẽ mang lại giá trị 18 cho hàm f, chuỗi 11 số 1 sẽ mang lại giá trị 29 cho hàm f ... điều này có thể đẩy việc tìm kiếm theo hướng sai, về phía cực đại cục bộ. Như vậy, với bài toán có nhiều tối ưu cục bộ thì cơ hội để có được tối ưu toàn cục của giải thuật leo đồi là rất khó khăn.

1.4.2 Giải thuật luyện thép

Ta xét giải thuật luyện thép giải bài toán trên:

Procedure Luyenthep;

Begin

1. t:=0;
2. Khởi tạo nhiệt độ T
3. Chọn ngẫu nhiên chuỗi v_c làm chuỗi hiện hành
4. Tiến hóa v_c
5. **Lặp lại**

    Lặp lại

    +Chọn chuỗi v_n mới trong lân cận của v_c bằng cách thay một bít của v_c

    + Nếu $f(v_c)<f(v_n)$ thì $v_c:=v_n$

    Ngược lại thì

    nếu random[0,1)<exp $\{(f(v_n)-f(v_c))/T\}$ thì $v_c:=v_n$

    Cho đến khi (điều kiện dừng thỏa)

    + T:=g(T,t);

    + t:=t+1;

Cho đến khi (điều kiện dừng 2 thỏa);

End;

Hàm random[0,1) trả về một số ngẫu nhiên trong khoảng [0,1). Các điều kiện dừng, dùng để kiểm tra xem đã đạt "cân bằng nhiệt độ" chưa, nghĩa là phân bố xác suất của các chuỗi mới đã chọn có đạt đến phân bố Boltzman không. Tuy nhiên, trong một số cài đặt, vòng lặp chỉ được thực hiện k lần, k là tham số bổ sung của phương pháp.

Nhiệt độ T được hạ thấp theo từng bước (g(T,t)<T với mọi t). Giải thuật dừng khi T đạt một giá trị đủ nhỏ: tiêu chuẩn dừng và kiểm tra xem hệ thống đã "đông" lại chưa, nghĩa là, không thay đổi nào được chấp nhận nữa.

Giải thuật luyện thép có thể thoát khỏi tối ưu cục bộ, xét chuỗi sau: v_4=(111000000100110111001010100000) với 12 số 1, cho giá trị của $f(v_4)=18$, với v_4 là chuỗi khởi tạo, giải thuật leo đồi có thể tiến đến cực đại cục bộ

v_l=(000000000000000000000000000000)

Do một chuỗi 13 số 1 bất kỳ (nghĩa là, một bước"tiến đến" tối ưu toàn cục) cho giá trị 7 (nhỏ hơn 18). Mặt khác, giải thuật mô phỏng luyện thép có thể nhận một chuỗi có 13 số 1 làm chuỗi hiện hành mới với xác suất

$p = \exp\{(f(v_n)-f(v_c))/T\} = \exp\{(7-18)/T\}$ đối với một nhiệt độ nào đó, chẳng hạn T=20, khi đó ta sẽ có $p=e^{-11/20} = 0.57695$. Điều này có nghĩa là các cơ hội chấp nhận tốt hơn 50%.

1.4.3 Giải thuật di truyền

Giải thuật di truyền được xây dựng tương tự như giải thuật trong 1.3.1. Trong quá trình tiến hóa luôn luôn duy trì một quần thể các chuỗi. Xét hai chuỗi tương đối xấu:

v_5=(111110000001│110111001110100000)và

v_6=(00100│0000001│101110010101111111)

Mỗi chuỗi được lượng giá là 15(│11*15-150│=15, có thể sinh ra con tốt hơn nhiều nếu điểm lai rơi vào bất cứ vị trí nào giữa vị trí 5 và vị trí thứ 12.

v_7=(111110000001│101110010101111111) con mới được lượng giá bằng

$f(v_7)$= │11*19-150│=59 thích nghi hơn cha-mẹ.

Chương 2
CƠ SỞ TOÁN HỌC, CƠ CHẾ THỰC HIỆN & NGUYÊN LÝ HOẠT ĐỘNG CỦA GIẢI THUẬT DI TRUYỀN

2.1. Cơ sở toán học của GA

2.1.1. Một số khái niệm

- Một chuỗi S = (s_1, ... , s_n) trên bảng chữ cái {0, 1, *} gọi là một sơ đồ (nhị phân) độ dài n. Ký hiệu * được gọi là ký tự đại diện.

- Một chuỗi v = (v_1, ... , v_n) trên bảng chữ cái {0, 1} gọi là thoả sơ đồ S nếu và chỉ nếu nó trùng khớp với S tại các vị trí không là ký tự đại diện *, nghĩa là với $\forall i \in \{j \mid s_j \neq *\}$ thì $s_i = v_i$. Ký hiệu khi đó $v \in S$.

- Bậc của sơ đồ S là số các ký hiệu xác định (ký hiệu khác *) trong S :

$$O(S) = |\{ i \in \{1, ... , n\} \mid s_i \neq *\}|$$

- Khoảng cách giữa các vị trí xác định đầu tiên và cuối cùng gọi là chiều dài xác định sơ đồ S :

$$\delta(S) = \max \{i \mid s_i \neq *\} - \min \{i \mid s_i \neq *\}$$

Xét giải thuật di truyền tổng quát, cùng với phép chọn tỷ lệ và hàm thích nghi f tuỳ ý (song đã cố định).

- Số các cá thể thoả sơ đồ S tại thời điểm t được ký hiệu là

$$r_{S,t} = | B_t \cap S |$$

- Biểu thức $\overline{f}(t) = \frac{1}{N}\sum_{i=1}^{N} f(b_{i,t})$ cho độ thích nghi trung bình thu được tại thời điểm t. Với $f(b_{i,t})$ là đội thích nghi của cá thể thứ i tại thời điểm t. N là kích thước của quần thể.

- Ký hiệu $\overline{f}(S,t)$ chỉ độ thích nghi trung bình quan sát được của sơ đồ S tại thời điểm t

$$\overline{f}(S,t) = \frac{1}{r_{S,t}} \sum_{i \in \{ j \updownarrow b_{j,t \in S} \}} f(b_{i,t})$$

2.1.2. Định lý sơ đồ (Holland 1975)

a. Định lý : Với giải thuật di truyền, bất đẳng thức sau được duy trì với mỗi sơ đồ S (có độ dài n)

$$E[r_{S,t+1}] \geq r_{S,t} * \frac{\overline{f}(S,t)}{\overline{f}(t)} * (1 - p_c * \frac{\delta(S)}{n-1}) * (1 - p_m)^{O(S)} \qquad (2.4)$$

b. Hệ quả: Với giải thuật di truyền đã mô tả và toán tử chọn lọc theo bánh xe Roulette, bất đẳng thức sau được duy trì

$$E[r_{S,t+1}] \geq \frac{\overline{f}(S,t)}{\overline{f}(t)} * r_{S,t} * P_C(S) * P_M(S) \qquad (2.6)$$

với mỗi sơ đồ S, ở đây $P_C(S)$ là một hằng số chỉ phụ thuộc sơ đồ S và phép lai ghép; $P_M(S)$ là hằng số chỉ phụ thuộc sơ đồ S và phép đột biến.

Cụ thể chúng ta có các ước lượng sau :

$P_C(S) = 1 – p_c.(\delta(S)/(n-1))$	với lai ghép 1 điểm
$P_C(S) = 1 – p_c.(1-(1/2)^{O(S)})$	với lai ghép đồng dạng
$P_C(S) = 1 – p_c$	Các phương pháp lai ghép khác
$P_M(S) = (1 - P_M)^{O(S)}$	đột biến bitwise
$P_M(S) = 1 - P_M^{O(S)/n}$	đảo ngược 1 bit đơn
$P_M(S) = 1 - P_M$	đảo ngược bitwise
$P_M(S) = 1 - P_M.(\lvert S\rvert/2^n)$	chọn ngẫu nhiên

Một sơ đồ có độ thích nghi trên độ thích nghi trung bình của quần thể gọi là "trên trung bình". Từ định lý sơ đồ có thể thấy rằng các sơ đồ trên trung bình sẽ nhận được số cá thể trong quần thể kế tiếp tăng lên trong khi các sơ đồ "dưới trung bình" nhận số chuỗi giảm đi.

Từ công thức 2.4 có thể phát biểu định lý sơ đồ sau :

***c. Định lý sơ đồ* :** Các sơ đồ ngắn, bậc thấp, trên trung bình nhận số chuỗi phù hợp tăng theo luỹ thừa trong các thế hệ tiếp theo của giải thuật di truyền.

Có một số hạn chế của định lý sơ đồ : Thứ nhất, do đánh giá là bất đẳng thức, việc các chuỗi bị bỏ qua tăng và các chuỗi bị mất đi nên có nhiều thông tin bị bỏ qua. Việc chưa chính xác của bất đẳng thức còn hạn chế ở chỗ định lý sơ đồ dự báo việc biểu diễn sơ đồ riêng đối với việc tạo sinh phức tạp, kết quả dự báo trong một số trường hợp có thể bị lỗi hoặc có hại.

Thứ hai là độ thích nghi quan sát của sơ đồ S tại thời điểm t có thể thay đổi đột ngột như quần thể tập trung các mẫu mới trong một bộ phận con đặc biệt của sơ đồ. Do vậy độ thích nghi trung bình của các chuỗi trong một sơ đồ đặc biệt chỉ quan hệ với lần tạo sinh thứ nhất hay thứ hai.

Nhìn chung, định lý sơ đồ đã chuẩn bị một cận dưới cho chỉ một lần tạo sinh trong tương lai. Tuy nhiên nó không thể dự báo biểu diễn của sơ đồ S với các lần tạo sinh phức tạp.

2.1. 3. Giả thuyết khối kiến trúc

Giải thuật di truyền tìm việc thực hiện gần tối ưu thông qua việc đặt gần nhau các sơ đồ ngắn, bậc thấp, hiệu quả cao được gọi là các khối kiến trúc.

Ta thấy một quần thể kích cỡ pop_size với chiều dài L tạo ra ít nhất 2^L và nhiều nhất là pop_size*2^L sơ đồ. Một số sơ đồ được xử lý theo hướng có lợi, chúng sinh sản theo tốc độ luỹ thừa và không bị loại bỏ qua phép lai và đột biến (điều này có thể xảy ra với các sơ đồ có chiều dài xác định lớn và bậc cao). Cũng cần lưu ý thêm là đây chỉ là giả thuyết và được tin là đúng. Nó có thể bị vi phạm, chẳng hạn xét 2 sơ đồ ngắn bậc thấp:

S_1 = (1 1 1 * * * * * * * * *)

S_2 = (* * * * * * * * * 1 1)

là trên trung bình, song kết hợp chúng lại ít phù hợp hơn:

S_3 = (1 1 1 * * * * * * 1 1)

2.1.4. Mô hình Markov của giải thuật di truyền

Trong giải thuật di truyền, mỗi quần thể được tạo sinh có đặc trưng là chỉ phụ thuộc theo xác suất vào quần thể trước đó, do vậy GA có thể được mô hình hóa bởi một xích Markov. Các tính chất của quần thể hữu hạn, chẳng hạn như xu hướng di truyền có thể được khảo sát cẩn thận sử dụng xích Markov hữu hạn. Không giống như nhiều phương pháp mô phỏng khác, xích Markov có thể mô tả chính xác và chi tiết cho mỗi lớp đặc biệt. Ta xét mô hình xích Markov của giải thuật di truyền đơn giản :

a. Tính Markov

- Một hệ vật lý hoặc hệ trạng thái nào đó gọi là có tính Markov nếu sự tiến triển của hệ trong tương lai chỉ phụ thuộc vào hiện tại và độc lập với quá khứ. Chẳng hạn như sự tăng dân số.
- Ký hiệu E là không gian trạng thái của X(t); nếu X(t) có tính Markov và E là tập đếm được thì X(t) gọi là xích Markov. Nếu giá trị của thời điểm t là 0, 1, … thì ta nói xích Markov với thời gian rời rạc. Nếu $t \in [0, \infty]$ thì ta có xích Markov với thời gian liên tục.
- Định nghĩa: Ta nói X(t) có tính Markov nếu:

  $P\{X(t_n+1) = j \mid X(t_0) = i_0, \dots, X(t_n) = i\} = P\{X(t_n+1) = j \mid X(t_n) = i\}$

  với bất kỳ $t_0 < t_1 < \dots < t_n < t_{n+1} < \dots$; $i_0, \dots, i_n, j \in E$
- Đặt $p(s,i,t,j) = P\{X(t) = j \mid X(s) = i\}$ $(s<t)$ là xác suất có điều kiện để hệ tại thời điểm s ở trạng thái i, đến thời điểm t chuyển đến trạng thái j và gọi là xác suất chuyển của hệ. Nếu xác suất này chỉ phụ thuộc vào t-s hay $p(s,i,t,j) = p(s+h, i, t+h, j)$ thì hệ là thuần nhất theo thời gian.
- Ký hiệu $p_{ij} = P\{X_{n+1} = j \mid X_n = i\}$ và ma trận $[p_{ij}]$ gọi là ma trận xác suất chuyển sau 1 bước. Theo công thức xác suất đầy đủ ta có

  $0 \le p_{ij} \le 1$ và $\Sigma_j\, p_{ij} = 1$ (ma trận ngẫu nhiên)
- Xác suất chuyển sau n bước :

  $p_{ij}^{(n)} = P\{X_{n+m} = j \mid X_m = i\} = P\{X_n = j \mid X_0 = i\}$

  Khi đó $p_{ij}^{(1)} = p_{ij}$. Quy ước $p_{ij}^{(0)} = 1$ (i=j) và là 0 với i<> j

  Đặt $P^n = (p_{ij}^{(n)})$ là ma trận xác suất chuyển sau n bước.
- Ta có phương trình Chapman-Kolmogorov

$$p_{ij}^{(n+m)} = \sum_{k \in E} p_{ik}^{n} p_{jk}^{n}$$

b.Một số kết quả

- Phân phối của hệ tại thời điểm n cho bởi công thức
  $p_j^{(n)} = P(X_n = j)$; n = 0, 1, 2, …; $j \in E$
  Đặt $\Pi^{(n)} = \{p_j^{(n)} ; j \in E)$ và gọi $\Pi = \Pi^{(0)}$ là phân phối ban đầu của hệ.
- Phân phối ban đầu gọi là dừng nếu $\Pi^{(n)}$ không phụ thuộc vào n nghĩa là $\Pi = \Pi^{(n)}$.
- Một mô hình xích Markov rời rạc và thuần nhất là bộ ba (X_n, Π, P) trong đó X_n là dãy đại lượng ngẫu nhiên rời rạc; Π là phân phối ban đầu; P là ma trận xác suất chuyển.
- Vấn đề là phân phối dừng có tồn tại không? có duy nhất không? cách tìm nó?

b1. Xích Markov trong GA

Xét bài toán: *Cho một hàm $f : J \to R^+$ và J_L là tập rời rạc hóa J. Cần tìm một phần tử $i \in J_L$ sao cho $f(i) = max\{f(j) \mid j \in J_L\}$*

Gọi r là số cá thể hợp lệ thì $r = |J_L| = \alpha^L$, trong đó α là số phần tử của bảng ký hiệu sử dụng rời rạc hóa. Chẳng hạn với mã hóa nhị phân thì $\alpha = 2$.

Gọi tập tất cả các quần thể hợp lệ là S và giả thiết ta xét quần thể với kích cỡ cố định M (M = |k| với $k \in S$). Khi đó số các quần thể hợp lệ là:

$$N = |S| = C_{M+r-1}^{r-1} \quad \text{(số các tổ hợp chập r-1 của M+r-1).}$$

Ta xét giải thuật di truyền đơn giản với các toán tử như sau:

- Toán tử chọn lọc tỷ lệ, chọn cá thể i trong quần thể k với xác suất:

$$p_{ik} = \frac{f(i)*Z(i,k)}{\sum_{h=0}^{r} f(h)*Z(h,k)}$$

  trong đó Z(i,k) là biến cố xảy ra khi cá thể i thuộc vào quần thể k.
- Toán tử lai ghép một điểm.
- Toán tử đột biến là phép đảo bit tại vị trí đột biến.

Với giải thuật này, GA có thể được mô hình hóa như một xích Markov thời gian rời rạc, mỗi quần thể xem như một trạng thái của xích. Rõ ràng là trạng thái hiện tại chỉ phụ thuộc vào một trạng thái ngay trước nó, vì nếu ký hiệu X_n là quần thể tại lần tạo sinh thứ n thì với n = 1,2,... ta có:

$$P(X_n = k \mid X_0 = k_0 , X_1 = k_1, ..., X_{n-1} = k_{n-1}) = P(X_n = k \mid X_{n-1} = k_{n-1})$$

nghĩa là X_n thỏa tính chất của xích Markov. Ký hiệu ma trận chuyển của xích Markov này là Q.

Định nghĩa 1: Giả sử $B_t = max\{f(i) \mid Z(i, X_i) \neq 0, i = 0,1, ..., r-1\}$ là chuỗi biến ngẫu nhiên của độ thích nghi cao nhất trong một quần thể biểu diễn trạng thái X_t tại bước thứ t. Giải thuật di truyền là *hội tụ đến tối ưu toàn cục* nếu và chỉ nếu $\lim_{t\to\infty} P(B_t = f^*) = 1$, ở đây f* = max{f(j) | j $\in$ J_L } là lời giải tối ưu của bài toán nêu trên.

Do các toán tử của GA độc lập với nhau nên có thể biểu diễn ma trận chuyển là Q = SCM trong đó S, C, M theo thứ tự là ma trận chuyển của các phép chọn lọc, lai ghép và đột biến.

Định nghĩa .2: *Khoảng cách Hamming* giữa hai chuỗi bit là số vị trí bit không trùng nhau của hai chuỗi đó (chẳng hạn H('0100', '0001') = 2). Đặt H_{ij} là tổng khoảng cách Hamming giữa các cá thể trong quần thể i và j:

$$H_{ij} = \sum_{h=0}^{M-1} H(i_h, j_h)$$

ở đây i_h là một cá thể trong quần thể i, j_h là một cá thể trong quần thể j với h=0, ..., M-1; i,j = 1, ..., N và $H(i_h, j_h)$ là khoảng cách Hamming giữa i_h và j_h.

Định nghĩa 3 : Ma trận A gọi là nguyên sơ nếu tồn tại số tự nhiên k sao cho ma trận A^k là ma trận dương (ma trận mà mọi phần tử đều là số dương).

Bổ đề 3.1 : Ma trận chuyển của giải thuật di truyền đơn giản với xác suất đột biến p_m $\in$ (0, 1), sử dụng lai ghép và chọn lọc tỷ lệ là ma trận nguyên sơ.

Với các ký hiệu trên, điều kiện hội tụ của GA có thể phát biểu là:

Giải thuật di truyền hội tụ tới tối ưu toàn cục nếu và chỉ nếu $\lim_{t\to\infty} X_t \in K$

Ma trận chuyển Q^+ đối với chiến lược ưu tú sửa đổi như sau:

$$\begin{bmatrix} Q_0 & 0 & 0 & 0 \\ 0 & Q_1 & 0 & 0 \\ & & \cdots & \\ 0 & 0 & 0 & Q_{r-1} \end{bmatrix}$$

Q^+ gồm r ma trận con Q_0, Q_1, ..., Q_{r-1} , mỗi ma trận có phần trên đường chéo chính bằng 0. Kích cỡ của Q_i là N(i)xN(i). Có thể thấy N(i) tính được là số tổ hợp chập (r-i-1) của (r-i+M-2).

Dựa trên các ký hiệu trên, Suzuki đã đưa ra công thức tính xác suất chuyển từ trạng thái k đến trạng thái v là:

$$q_{kv} = \begin{cases} (M-1)! \prod_{j=0}^{r-1} \dfrac{1}{Y(j,v)!} r(j,k)^{Y(j,v)} & i^*(k) \le i^*(v) \\ 0 & i^*(k) > i^*(v) \end{cases}$$

trong đó

$$Y(j,k) = \begin{cases} Z(j,k) & j \ne i^*(k) \\ Z(j,k) - 1 & j = i^*(k) \end{cases}$$

Với Z(j,k) là xác suất để cá thể j thuộc vào quần thể k; r(j,k) là xác suất để cá thể j được tạo lập bởi các toán tử di truyền từ quần thể k. Nếu $p_m > 0$ thì r(j,k) > 0. Bởi vậy

$q_{kv} > 0$ với k, v $\in$ K vì i*(k) = i*(v). Điều này cũng có nghĩa là Q_0 là ma trận dương và là nguyên sơ. Tuy nhiên cấu trúc của ma trận không thể suy ra nhờ trực giác bằng sự nghiên cứu mỗi toán tử di truyền. Toán tử chọn lọc không thể tạo cá thể mới và cũng không thể làm tăng i*(k). Toán tử lai ghép và đột biến đều tạo cá thể mới và có xác suất dương khi chuyển đến quần thể mới có cá thể cao hơn i*(k).

Định lý 3.2: Giải thuật di truyền với chiến lược ưu tú sửa đổi như trên và $p_m \in (0, 1)$ hội tụ đến tối ưu toàn cục.

Chứng minh: Từ những phân tích trên, ta có Q^+ là không phân ly được và Q_0 là ma trận nguyên sơ. Theo lý thuyết xích Markov, ta có

$$Q_\infty^+ = \lim_{n\to\infty}(Q^+)^k = \begin{pmatrix} \lim\limits_{k\to\infty} Q_0^k & 0 \\ R^\infty & 0 \end{pmatrix}$$

trong đó R^∞ là ma trận ngẫu nhiên chứa các xác suất chuyển từ trạng thái không tối ưu toàn cục đến trạng thái tối ưu toàn cục. Điều này cho thấy xác suất sống sót của trạng thái không tối ưu toàn cục có khuynh hướng giảm dần đến 0 còn xác suất sống sót của trạng thái tối ưu toàn cục hội tụ về 1.

Như vậy, định lý này cho thấy giải thuật di truyền có khả năng hội tụ về tối ưu toàn cục sau quá trình tiến hoá.

Việc mô hình GA qua xích Markov có ý nghĩa quan trọng. Nó vừa chứng tỏ được sự hội tụ của giải thuật một cách chính xác, vừa là mô hình toán học để có thể tính toán chi tiết các yếu tố ảnh hưởng đến khả năng và tốc độ hội tụ của giải thuật. Tuy nhiên, thực hiện thuật toán qua xích Markov sẽ gặp khó khăn rất lớn do không gian trạng thái cần xét đến là rất lớn. Vì vậy mô hình này thường chỉ được phát triển về mặt lý luận.

2.2. Cơ chế thực hiện

Giả sử, tìm cực đại hàm k biến $f(x_1, x_2, ..., x_k)$: $R^k \to R$, mỗi biễn x_i nhận giá trị trong miền $D_i=[a_i, b_i] \subseteq R$ và $f(x_1, x_2, ..., x_k) > 0$ với $\forall x_i \in D_i$. Tối ưu hóa hàm f với độ chính xác: 6 số lẻ đối với giá trị của các biến.

Để đạt được độ chính xác như trên thì mỗi miền D_i được phân cắt thành $(b_i-a_i) \times 10^6$ miền con bằng nhau. Gọi m_i là số nguyên nhỏ nhất sao cho:

$(b_i-a_i) \times 10^6 \le 2^{m_i} - 1$

Mỗi biến x_i được biểu diễn bằng một chuỗi nhị phân có chiều dài m_i, với biểu diễn này rõ ràng thỏa mãn điều kiện về độ chính xác yêu cầu. Công thức sau tính giá trị thập phân của mỗi chuỗi nhị phân biểu diễn biến x_i

$$x_i = a_i + decimal(111001...001_2).\frac{b_i - a_i}{2^{m_i} - 1}$$ trong đó decimal(chuỗi $_2$) cho biết giá trị thập phân của chuỗi nhị phân đó.

Mỗi nhiễm sắc thể (một lời giải) được biểu diễn bằng chuỗi nhị phân có chiều dài $m = \sum_{i=1}^{k} m_i$; m_1 bit đầu tiên biễu diễn các giá trị trong khoảng [a_1, b_1]; m_2 bit kế tiếp biểu diễn các giá trị trong khoảng [a_2, b_2]; ...; nhóm m_k bit cuối cùng biểu diễn các giá trị trong khoảng [a_k, b_k];

Để khởi tạo quần thể ta cần tạo ra pop-size nhiễm sắc thể ngẫu nhiên theo từng bit, tuy nhiên cần dùng thông tin về sự phân phối để khởi tạo quần thể ban đầu.

Tiếp theo, trong mỗi thế hệ ta lượng giá từng nhiễm sắc thể (tính giá trị hàm f trên các chuỗi nhị phân đã được giải mã), chọn quần thể mới thỏa mãn phân bố xác suất dựa trên độ thích nghi và thực hiện các phép đột biến và lai để tạo ra các cá thể của thế hệ mới. Sau một số thế hệ, khi không còn cải thiện thêm được gì nữa, nhiễm sắc thể tốt nhất sẽ được xem như lời giải của bài toán tối ưu 9thường là toàn cục. Thông thường, giải thuật di truyền được dừng lại sau một số bước lặp cố định tùy thuộc điều kiện về tốc độ và tài nguyên máy tính.

Với tiến trình chọn lọc (chọn quần thể mới thỏa phân bố xác suất dựa trên các độ thích nghi) ta dùng bánh xe quay Rulet với các rãnh được định kích thước theo độ thích nghi, bánh xe Rulet được xây dựng như sau:
(với giả thiết các độ thích nghi đều dương, trong trường hợp ngược lại ta có thể dùng một số phép biến đổi tương ứng để định lại tỷ lệ sao cho các độ thích nghi đều dương)

+ Tính độ thích nghi eval(v_i) của mỗi nhiễm sắc thể v_i (i=1...pop-size)

+ Tìm tổng giá trị thích nghi toàn quần thể: F= $\sum_{i=1}^{pop-size} eval(v_i)$

+ Tính xác suất chọ p_i cho mỗi nhiễm sắc thể v_i, (i=1...pop-size), $p_i = \frac{eval(v_i)}{F}$

+ Tính vị trí xác suất q_i của mỗi nhiễm sắc thể v_i, (i=1...pop-size), $q_i = \sum_{j=1}^{i} p_j$

Tiến trình chọn lọc được thực hiện bằng cách quay bánh xe Rulet pop-size lần, mỗi lần chọn một nhiễm sắc thể từ quần thể hiện hành vào quần thể mới theo cách:

+ Phát sinh ngẫu nhiên một số r trong khoảng [0..1]

+ Nếu $r<q_1$ thì chọn nhiễm sắc thể đầu tiên (v_1); ngược lại thì chọn nhiễm sắc thể thứ i, v_i ($2\leq i\leq$ pop-size) sao cho $q_{i-1}< r\leq q_i$

Có thể sẽ có một số nhiễm sắc thể được chọn nhiều lần. Điều này là phù hợp với lý thuyết sơ đồ: Các nhiễm sắc thể tốt nhất có nhiều bản sao hơn, các nhiễm sắc thể trung bình không thay đổi, các nhiễm sắc thể kém nhất thì chết đi.

Tiếp theo, ta dùng các phép toán di truyền: Kết hợp, lai vào các cá thể trong quần thể mới vừa được chọn từ quần thể cũ. Một trong các tham số của hệ di truyền là xác suất lai p_c. Xác suất này cho ta số nhiễm sắc thể pop-size x p_c , các nhiễm sắc thể này được dùng trong tiến trình lai tạo. Ta tiến hành theo cách sau đây:

Đối với mỗi nhiễm sắc thể trong quần thể mới:

+ Phát sinh ngẫu nhiên một số r trong khoảng [0..1]

+ Nếu $r<p_c$, hãy chọn nhiễm sắc thể đó để lai tạo

Sau đó, ta ghép đôi các nhiễm sắc thể đã chọn được một cách ngẫu nhiên: đối với mỗi cặp nhiễm sắc thể được ghép đôi, ta phát sinh ngẫu nhiên một số nguyên pos trong khoảng [1..m-1] (m là tổng chiều dài – số bít của một nhiễm sắc thể). Số poscho biết vị trí của điểm lai. Xét 2 nhiễm sắc thể:$(b_1b_2...b_{pos}b_{pos+1}...b_m)$ và $(c_1c_2...c_{pos}c_{pos+1}...c_m)$ được thay bằng một cặp con của chúng :$(b_1b_2...b_{pos}c_{pos+1}...c_m)$, $(c_1c_2...c_{pos}b_{pos+1}...b_m)$

Phép đột biến được thực hiện trên cơ sở từng bít. Tham số p_m của hệ thống di truyền, cho ta số bít đột biến p_m x m x pop-size. Mỗi bit (trong tất cả các nhiễm sắc thể trong quần thể) có cơ hội bị đột biến như nhau, nghĩa là đổi từ 0 thành 1 hoặc ngược lại, vì vậy ta tiến hành theo cách:

Đối với mỗi nhiễm sắc thể trong quần thể hiện hành (sau khi lai) và đối với mỗi bit trong nhiễm sắc thể:

+ Phát sinh ngẫu nhiên một số r trong khoảng [0..1]

+ Nếu $r<p_m$, hãy đột biến bit đó

Sau quá trình chọn lọc, lai và đột biến, quần thể mới được lượng giá. Lượng giá này được dùng để xây dựng phân bố xác suất (cho tiếp trình lựa chọn kế tiếp), nghĩa là xây dựng lại bánh xe Rulet với các rãnh được định kích thước theo các giá trị thích nghi hiện hành. Phần còn lại của tiến hóa là lặp lại chu trình của những bước trên.

Ví dụ:

Cực đại hóa hàm $f(x_1, x_2)=21.5+x_1\sin(4\pi x_1)+x_2\sin(20\pi x_2)$ với $-3.0\le x_1\le 12.1$ và $-4.1\le x_2\le 5.8$

Giả sử cần tính chính xác đến 4 số lẻ đối với mỗi biến. Miền của biến x_1 có chiều dài là 12.1-(-3)=15.1, do đó điều kiện chính xác đòi hỏi đoạn [-3,12.1] được chia thành các khoảng có kích thước bằng nhau, số khoảng là 15.1 x 10^4 khoảng, điều này có nghĩa cần 18 bít làm phần đầu tiên của nhiễm sắc thể: $2^{17}\le 151000\le 2^{18}$.

Miền của biến x_2 có chiều dài 5.8-4.1=1.7 nên cần chia [4.1,5.8] thành 1.7 x 10^4 khoảng có kích thước bằng nhau, điều này có nghĩa là cần 15 bít làm phần đầu tiên của nhiễm sắc thể: $2^{14}\le 17000\le 2^{15}$. Như vậy, chiều dài toàn bộ NST là m=18+15 =33 bít (18 bít đầu tiên mã hóa x_1, 15 bít còn lại mã hóa x_2).

Ví dụ: Xét NST v

0	1	0	0	0	1	0	0	1	0	1	1	0	1	0	0	0	0	1	1	1	1	1	0	0	1	0	1	0	0	0	1	0	v

18 bít đầu tiên biểu diễn

0	1	0	0	0	1	0	0	1	0	1	1	0	1	0	0	0	0

x_1=-3.0 + decimal$(010001001011010000)_2$ x (12.1-(-3.0))/(2^{18}-1)=

-3.0+70352x 15.1/2262143=-3.0 + 4.052426= 1.052426

15 bít kế tiếp

1	1	1	1	1	0	0	1	0	1	0	0	0	1	0

biểu diễn x_2=4.1 + decimal$(111110010100010)_2$ x (5.8-4.1)/(2^{15}-1)=

4.1+31906 x 1.7/32767=5.755330

Như vậy, NST v tương ứng với (x_1, x_2)=(1.052426, 5.755330). Độ thích nghi của NST này là f((x_1, x_2)=20.252640.

Để cực đại hóa hàm f bằng giải thuật di truyền ta cần tạo ra một quần thể có pop_size=20 NST, cả 33 bít trong tất cả các NST đều được khởi tạo ngẫu nhiên. Giả sử có quần thể gồm 20 NST (từ v_1 đến v_{20})

1	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	v_1
1	1	1	0	0	0	1	0	0	1	0	0	1	1	0	1	1	1	0	0	1	0	1	0	1	0	0	1		1	0	1	0	v_2
0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	1	0	1	1	1	0	1	v_3
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_4
0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1	0	0	0	1	0	1	v_5
0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	1	1	v_6
0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	v_7
1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	v_8
0	1	1	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	v_9
0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	v_{10}
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
1	1	0	1	0	0	0	1	0	1	1	1	1	0	1	1	0	1	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0	v_{12}
1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v_{13}
0	1	0	0	1	0	0	1	1	0	0	0	0	0	1	0	1	0	1	0	0	1	1	1	1	0	0	1	0	1	0	0	1	v_{14}
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}
1	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	1	v_{16}
0	1	1	0	1	0	1	1	1	1	1	1	0	0	1	1	1	1	0	1	0	0	0	1	1	0	1	1	1	1	1	0	1	v_{17}
0	1	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1	0	1	0	0	1	1	1	1	1	0	1	0	1	1	0	1	v_{18}
0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	1	0	0	v_{19}
1	0	1	1	1	0	0	1	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	v_{20}

Ta tính độ thích nghi (lượng giá) cho từng NST có kết quả như sau:

x_1	x_2	$eval(v_i)=f(x_1,x_2)$	x_1	x_2	$eval(v_i)=f(x_1,x_2)$
6.084492	5.652242	26.019600	3.130078	4.996097	23.410669
10.348434	4.380264	7.580015	9.356179	4.239457	15.011619
-2.516603	4.390381	19.626329	11.134646	5.378671	27.316702
5.278638	5.593460	17.406725	1.335944	5.151378	19.876294
-1.255173	4.734458	25.341160	11.089025	5.054515	30.060205
-1.811725	4.391937	18.100417	9.211598	4.993762	23.967227
-0.991471	5.680258	16.020812	3.367514	4.571343	13.696165
4.910618	4.703018	17.959701	3.843020	5.158226	15.414128
0.795406	5.381472	16.127799	-1.746635	5.395584	20.095903
-2.554851	4.793707	21.278435	7.935998	4.757338	13.666916

Qua bảng lượng giá trên ta thấy NST v_{15} là mạnh nhất, NST v_2 là yếu nhất.

Tổng độ thích ghi của quần thể là: $F = \sum_{i=1}^{20} eval(v_i) = 387.776822$

+ Xác suất chọn lọc p_i của mỗi NST v_i (i=1..20) được tính là:

p_i	$p_i=eval(v_i)/F$	p_i	$p_i=eval(v_i)/F$
p_1	0.067099	p_{11}	0.060372
p_2	0.0195547	p_{12}	0.038712
p_3	0.050355	p_{13}	0.070444
p_4	0.044889	p_{14}	0.051257
p_5	0.065350	p_{15}	0.077519
p_6	0.046677	p_{16}	0.061549
p_7	0.0413315	p_{17}	0.0353320
p_8	0.046315	p_{18}	0.039750
p_9	0.041590	p_{19}	0.051823
p_{10}	0.054873	p_{20}	0.035244

+ Tính các vị trí xác suất q_i của mỗi NST $q_i = \sum_{j=1}^{i} p_j$ i=1..20

q_i	$q_i = \sum_{j=1}^{i} p_j$	q_i	$q_i = \sum_{j=1}^{i} p_j$
q_1	0.067099	q_{11}	0.538381
q_2	0.086647	q_{12}	0.577093
q_3	0.137001	q_{13}	0.647537

q_4	0.181890	q_{14}	0.698794
q_5	0.247240	q_{15}	0.776314
q_6	0.293917	q_{16}	0.837863
q_7	0.335232	q_{17}	0.873182
q_8	0.381546	q_{18}	0.912932
q_9	0.423137	q_{19}	0.964756
q_{10}	0.478009	q_{20}	1.000000

+ Sau đó, ta quay bánh xe Rulét 20 lần, mỗi lần chọn một NST cho quần thể mới. Giả sử thứ tự ngẫu nhiên của 20 số trong [0,1] được phát sinh là:

1	0.513870	11	0.703899
2	0.175741	12	0.398647
3	0.308652	13	0.277226
4	0.534534	14	0.368071
5	0.947628	15	0.983437
6	0.171736	16	0.005398
7	0.702231	17	0.765682
8	0.226431	18	0.646473
9	0.494773	19	0.767139
10	0.424720	20	0.780237

Số đầu tiên r=0.513870 lớn hơn q_{10} và nhỏ hơn q_{11}, nghĩa là NST v_{11} được chọn vào quần thể mới. Số thứ hai là 0.175741 lớn hơn q_3 và nhỏ hơn q_4, nghĩa là NST v_4 được chọn vào quần thể mới... Như vậy quần thể mới gồm các NST sau:

0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_4
0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	v_7
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	1	0	0	v_{19}
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_4
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}
0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1	0	0	0	1	0	1	v_5
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	1	0	1	1	1	0	1	v_3
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}

0	1	1	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	v_9
0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	1	1	v_6
1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	v_8
1	0	1	1	1	0	0	1	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	v_{20}
1	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	v_1
0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	v_{10}
1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v_{13}
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}
1	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	1	v_{16}

Tiếp theo ta thực hiện phép kết hợp, lai cho những cá thể trong quần thể (v_i'). Xác suất lai là p_c=0.25, nghĩa là 5/20 NST sẽ tham gia lai tạo. Ta sẽ phát sinh ngẫu nhiên một số r trong khoảng [0,1], nếu r<0.25 ta chọn NST cho trước để lai tạo , giả sử thứ tự các số ngẫu nhiên là:

1	0.822951	11	0.031523
2	0.151932	12	0.869921
3	0.625477	13	0.166525
4	0.314685	14	0.574520
5	0.346901	15	0.758400
6	0.917204	16	0.581893
7	0.519760	17	0.389248
8	0.401154	18	0.200232
9	0.606758	19	0.355635
10	0.785402	20	0.826927

Như vậy NST v'_2, v'_{11}, v'_{13}, v'_{18} được chọn để lai tạo (nếu số NST được chọn là chẵn ta có thể ghép thành cặp dễ dàng, nếu số NST được chọn là số lẻ ta có thể lấy thêm một NST ngoại hay bỏ bớt một NST một cách ngẫu nhiên). Ta kết hợp lai một cách ngẫu nhiên giữa v'_2 và v'_{11} và cặp v'_{13} và v'_{18}. Đối với một cặp ta phát sinh một số pos thuộc khoảng [1..32], (32+1 là tổng chiều dài hay số bit trong một NST), số pos cho biết vị trí của điểm lai.

Cặp đầu tiên : v'_2

1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v'_2

V'_{11}

1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v'_{11}

Giả sử pos=9, các NST này bị cắt sau bít thứ 9 và được thay bằng cặp cặp con của chúng:

1	0	0	0	1	1	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v''_{2}

1	1	1	0	1	1	1	0	1	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v''_{11}

Cặp NST thứ 2 là

0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	1	1	v'_{13}

1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v'_{18}

Số phát sinh pos=20, các NST này được thay bằng cặp con của chúng:

0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	0	0	1	0	0	0	1	1	0	v''_{13}

1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1	1	1	1	1	0	1	1		$v''_{1'8}$

Tiếp tục ta sẽ thu được quần thể mới:

0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_1
1	0	0	0	1	1	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_2
0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	v_3
0	1	1	0	0	1	1	1	1	1	1	0	**0**	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_4
0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	1	0	0	v_5
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_6
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_7
0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1	0	0	0	1	0	1	v_8
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_9
0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	1	0	1	1	1	0	1	v_{10}
1	1	1	0	1	1	1	0	1	1	0	1	1	0	1	0	0	1	**1**	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_{11}
0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	v_{12}

0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	0	**0**	0	0	0	0	1	0	0	0	1	1	**0**	v_{13}
1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	v_{14}
1	0	1	1	1	0	0	1	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	v_{15}
1	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	v_{16}
0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	v_{17}
1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	1	0	1	0	1	1	1	1	1	1	0	1	1	v_{18}
1	1	1	0	1	1	1	0	**1**	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{19}
1	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	1	v_{20}

Tiếp theo, ta xét phép đột biến, được thực hiện dựa trên cơ sở từng bít. Xác suất đột biến p_m=0.01, vì thế trung bình sẽ có 1/100 số bít sẽ qua đột biến. Có 660 bit(m x pos-size=33 x 20) trong toàn quần thể, như vậy sẽ có 6.6 bít đột biến mỗi thế hệ. Mỗi bit có cơ hội đột biến ngang nhau, do đó đối với mỗi bit trong quần thể, ta phát sinh một số ngẫu nhiên r trong khoảng [0,1], nếu r<0.01 thì ta sẽ đột biến bit này.

Ta phải phát sinh 660 số ngẫu nhiên, giả sử có 5 trong số 660 số này nhỏ hơn 0.01, ví trí bít và số ngẫu nhiên được cho là:

Vị trí bít	Số ngẫu nhiên
112	0.000213
349	0.009945
418	0.008809
429	0.005425
602	0.002835

NST, vị trí của bit bị đột biến tương ứng với 5 vị trí bít trên

Vị trí bít	Số NST	Số bít trong NST
112	4	13
349	11	19
418	13	22
429	13	33
602	19	8

Điều này có nghĩa là 4 NST chịu ảnh hưởng của phép đột biến, trong đó có NST thứ 13, có 2 bít bị thay đổi.

Quần thể cuối cùng thu được là: (các bít đột biến được tô đậm)

0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_1
1	0	0	0	1	1	0	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_2
0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	v_3
0	1	1	0	0	1	1	1	1	1	1	0	**0**	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_4
0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	1	0	0	v_5
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_6
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_7
0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1	0	0	0	1	0	1	v_8
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_9
0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	1	0	1	1	1	0	1	v_{10}
1	1	1	0	1	1	1	0	1	1	0	1	1	0	1	0	0	1	**0**	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_{11}
0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	v_{12}
0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	0	**1**	0	0	0	0	1	0	0	0	1	1	**1**	v_{13}
1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	v_{14}
1	0	1	1	1	0	0	1	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	v_{15}
1	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	v_{16}
0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	v_{17}
1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	1	0	1	0	1	1	1	1	1	1	0	1	1	v_{18}
1	1	1	0	1	1	1	0	**0**	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{19}
1	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	1	v_{20}

Như vậy, ta vừa xét một bước lặp (một thế hệ) của thủ tục di truyền. Ta xét các kết quả của tiến trình tiến hóa quần thể mới, ta giải mã từng NST và tính các giá trị của hàm thích nghi từ các giá trị (x_1, x_2) vừa được giải mã. Ta có kết quả:

x_1	x_2	$eval(v_i)=f(x_1,x_2)$	x_1	x_2	$eval(v_i)=f(x_1,x_2)$
3.130078	4.996097	23.410669	11.088621	4.743434	33.351874
5.279042	5.054515	18.201083	0.795406	5.381472	16.127799
-0.991471	5.680258	16.020812	-1.811725	4.209937	22.692462
3.128235	4.996097	23.412613	4.910618	6.703018	17.959701
-1.746635	5.395584	20.095903	7.935998	4.757388	13.666916
5.278638	5.593460	17.406725	6.084492	5.652242	26.019600

11.089025	5.054515	30.060205	-2.554851	4.793707	21.278435
-1.255173	4.734458	25.341160	11.134646	5.666976	27.591064
3.130078	4.996097	23.410669	11.059532	5.054515	27.608441
-2.516603	4.390381	19.526329	9.211598	4.993726	23.867227

Như vậy tổng độ thích nghi F của quần thể mới là 447.049688 cao hơn độ thích nghi của quần thể trước (387.776822). NST tốt nhất v_{11} có độ thích nghi 33.351874 tốt hơn NST tốt nhất v_{15} của quần thể trước (30.060205)

Các tiến trình được lặp lại, các phép toán di truyền được thực hiện, lượng giá thế hệ kế tiếp...

2.3. Nguyên lý hoạt động của giải thuật di truyền

2.3.1. Nguyên lý hoạt động của phép chọn

1. Đặt ξ(S, t) là số chuỗi trong quần thể, ở thế hệ thứ t phù hợp với sơ đồ S. Đối với sơ đồ:

S_0=(*****111***************************)

Thì ξ(S_0, t)=3 vì có 3 chuỗi v_{13}, v_{15}, và v_{16} phù hợp với sơ đồ S_0. Ta có bậc của sơ đồ S_0 o(S_0)=3 và chiều dài xác định của nó δ(S_0)=7-5=2

2. Gọi eval(S,t) là độ thích nghi của sơ đồ S ở thế hệ t. Giả sử có p chuỗi {v_{i1}, v_{i2}, ...,v_{ip}} trong quần thể phù hợp với sơ đồ S vào thời điểm t thì:

$$eval(S,t)=\frac{\sum_{j=1}^{p} eval(v_{ij})}{p}$$

Trong bước chọn lọc, một quần thể trung gian được tạo ra gồm pop_size=20 các chuỗi được chọn ra từ quần thể hiện hành. Các chuỗi được chọn dựa vào độ thích nghi của nó và được chép vào quần thể thế hệ mới. Chuỗi v_i có xác suất được chọn là p_i=eval(v_i)/F(t) với f(t) là tổng thích nghi của toàn quần thể vào thời điểm t

$$F(t)=\sum_{i=1}^{20} eval(v_i)$$

Sau bước chọn lọc, ta có ξ(S, t+1) chuỗi phù hợp với sơ đồ S, bởi vì:

1. Với một chuỗi phù hợp
2. Ở thế hệ t, số chuỗi phù hợp với sơ đồ S là ξ(S, t) và
3. Chọn trong pop-size chuỗi

Do đó ξ(S, t+1)= ξ(S, t)*pop_size*eval(S,t)/F(t)

Với $\overline{F(t)} = F(t) / pop\_size$ là độ thích nghi trung bình của quần thể, công thức trên có thể viết lại là: ξ(S, t+1)= ξ(S, t)*eval(S,t)/$\overline{F(t)}$ (2.1) điều này có thể hiểu là, số chuỗi trong quần thể tăng bằng với tỉ lệ độ thích nghi của sơ đồ với độ thích nghi trung bình của quần thể (nghĩa là, sơ đồ trên trung bình sẽ nhận được thêm số chuỗi trong quần thể thế hệ kế tiếp, sơ đồ dưới trung bình nhận số chuỗi giảm đi, còn sơ đồ trung bình vẫn giữ nguyên mức. Với luật này ta có hệ quả, Nếu sơ đồ S vẫn giữ trên trung bình ε% : eval(S,t)= $\overline{F(t)}$ +ε* $\overline{F(t)}$ thì $\xi(S,t)= \xi(S, 0)(1+\varepsilon)^t$ và

ε= (eval(S,t)- $\overline{F(t)}$)/ $\overline{F(t)}$ (ε>0 đối với các sơ đồ trên trung bình và ε<0 đối với các sơ đồ dưới trung bình). Như vậy, một sơ đồ trên trung bình không những nhận số chuỗi tăng lên trong thế hệ kế tiếp mà nó còn tiếp tục nhận số chuỗi tăng theo lũy thừa trong các thế hệ kế tiếp.

Phương trình (2.1) được gọi là phương trình tăng trưởng sinh sản của sơ đồ.

Với sơ đồ S_0, có ba chuỗi v_{13}, v_{15} và v_{16} (vào thời điểm t) phù hợp với sơ đồ S_0, độ thích nghi eval(S_0) của sơ đồ là:

Eval(S_0,t)=(27.316702+30.060205+23.867227)/3=27.081378

Độ thích nghi trung bình của toàn thể quần thể là:

$\overline{F(t)} = \sum_{i=1}^{20} eval(v_i)$ /pop_size=387.776822/20=19.388841

Tỷ lệ thích nghi của sơ đồ S_0 đối với độ thích nghi trung bình của quần thể là:

eval(S_0,t)/ $\overline{F(t)}$ = 1.396751. Điều này có nghĩa là: Nếu sơ đồ S_0 trên trung bình thì nó nhận được số chuỗi tăng theo lũy thừa trong các thế hệ kế tiếp. Đặc biệt, nếu sơ đồ S_0 vẫn giữ trên trung bình do nhân với hằng số 1.396751 thì vào thời điểm (t+1) ta sẽ có 3*1.396751 =4.19 chuỗi phù hợp với S_0 (nghĩa là 4 hoặc 5 chuỗi), vào thời điểm (t+2) ta sẽ có $3*1.396751^2$ =5.8 chuỗi phù hợp với S_0(gần 6 chuỗi)...Như vậy, sơ đồ S_0 sẽ có tương lai khá tốt của không gian tìm kiếm và số mẫu đại diện của nó trong quần thể sẽ tăng theo lũy thừa.

Xét ví dụ với sơ đồ S_0. Trong quần thể vào thời điểm t, sơ đồ S_0 phù hợp với 3 chuỗi v_{13}, v_{15} và v_{16}, với tiến trình chọn lọc sử dụng cùng một quần thể để tạo ra quần thể mới, quần thể mới gồm các nhiễm sắc thể sau:

0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_4
0	0	1	0	0	0	1	0	0	0	0	0	1	1	0	1	0	1	1	1	1	0	1	1	0	1	1	1	1	1	0	1	1	v_7
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
0	0	0	1	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	1	1	0	0	0	1	1	0	0	v_{19}
1	0	0	0	1	1	0	0	0	1	0	1	1	0	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	0	0	1	0	v_4
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}

0	0	0	1	1	1	0	1	1	0	0	1	0	1	0	0	1	1	0	1	0	1	1	1	1	1	1	0	0	0	1	0	1	v_5
0	1	1	0	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	0	0	0	0	1	1	0	1	1	1	1	0	0	0	v_{11}
0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	1	0	1	1	1	0	1	1	1	0	1	v_3
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}
0	1	1	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	v_9
0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	1	1	v_6
1	0	0	0	0	1	1	0	0	0	0	1	1	1	0	1	0	0	0	1	0	1	1	0	1	0	1	1	0	0	1	1	1	v_8
1	0	1	1	1	0	0	1	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	0	v_{20}
1	0	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	v_1
0	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	1	0	1	0	0	0	0	1	1	1	0	1	1	v_{10}
1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v_{13}
1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v_{15}
1	1	0	0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	1	v_{16}

Tuy nhiên, riêng phép chọn chưa đem đến một kết quả mới nào đáng quan tâm từ không gian tìm kiếm, chọn lọc chỉ sao chép một số chuỗi để hình thành quần thể trung gian. Bước thứ hai của chu kỳ tiến hóa Tái kết hợp, có nhiệm vụ giới thiệu những cá thể mới trong quần thể, điều này được thực hiện bởi các phép toán di truyền: Lai và đột biến.

2.3.2. Nguyên lý hoạt động của Phép lai

Một chuỗi trong quần thể, ví dụ chuỗi v'_{18}

1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v_{13}

Phù hợp với tối đa 2^{33} sơ đồ, cụ thể là chuỗi trên phù hợp với 2 sơ đồ sau:

S_0=(****111**************************) và

S_1=(111****************************10)

Giả sử rằng, chuỗi này được chọn để thực hiện phép lai (v'_{18} được lai với v'_{13}) và vị trí lai phát sinh tại pos=20. Như vậy sơ đồ S_0 vẫn tồn tại, nghĩa là vẫn còn một con phù hợp với S_0, lý do là tại vị trí lai này bảo tồn chuỗi '111' trên các vị trí thứ 5, 6, 7 của chuỗi trong con.

Chuỗi con

1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	v'_{18}

Và

0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	1	0	1	0	1	1	1	1	1	1	0	1	1	v'_{13}

Sẽ sinh ra v''_{18} và v''_{13}

1	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1	1	1	0	1	0	1	1	1	1	1	1	0	1	1	v''_{18}
0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	v''_{13}

Tuy nhiên, sơ đồ S_1 có thể bị phá vỡ, vì có thể không có con nào phù hợp với nó với lý do là các vị trí cố định '111' ở đầu mẫu và các vị trí cố định '10' ở cuối được đặt vào con khác.

Như vậy, chiều dài xác định của sơ đồ đóng vai trò quan trọng trong xác suất bị loại bỏ hay tồn tại của sơ đồ. Lưu ý chiều dài xác định của sơ đồ S_1 là $\delta(S_0)$= 2 còn chiều dài xác định của sơ đồ S_1 là $\delta(S_1)$= 32 và các vị trí lai, trong số m-1 vị trí có cơ hội được chọn như nhau, điều này có nghĩa là xác suất bị loại của sơ đồ S là $p_d(S) = \frac{\delta(S)}{m-1}$ do đó xác suất tồn tại là $p_s(S) = 1 - \frac{\delta(s)}{m-1}$

Với ví dụ trên, các xác suất tồn tại và bị loại bỏ của các sơ đồ S_0 và S_1 là:

$p_d(S_0)=2/32$; $p_s(S_0)=30/32$; $p_d(S_1)=32/32=1$; $p_s(S_1)=0$;

Cần lưu ý thêm là chỉ có một số nhiễm sắc thể trải qua lai và xác suất lai là p_c, khi đó xác suất tồn tại của sơ đồ thực sự sẽ là: $p_s(S) = 1 - p_c\frac{\delta(s)}{m-1}$

Với ví dụ đang xét của sơ đồ S_0 ta xét với p_c=0.25 thì $p_s(S_0)$=1-0.25*2/32=63/64=0.984375

Như vậy khi đã chọn một vị trí lai trong số các vị trí cố định trong một sơ đồ, sơ đồ vẫn có thể tồn tại. Ví dụ, nếu 2 chuỗi v'_{18} và v'_{13} đều bắt đầu với '111' và tận cùng bằng '10' thì sơ đồ S_1 vẫn tồn tại (nhưng xác suất của hiện tượng này rất nhỏ), bởi vậy công thức xác suất tồn tại của sơ đồ được hiệu chỉnh là: $p_s(S) \geq 1 - p_c\frac{\delta(s)}{m-1}$

Tác động kết hợp của chọn lọc và lai cho một dạng mới phương trình tăng trưởng của sơ đồ sinh sản: $\xi(S, t+1) = \xi(S, t) * eval(S,t)/\overline{F(t)}\left[1 - p_c\frac{\delta(S)}{m-1}\right]$ (2.2)

Phương trình (2.2) cho biết kỳ vọng số chuỗi phù hợp với sơ đồ S trong thế hệ kế tiếp là hàm của số chuỗi đúng của sơ đồ, về độ thích nghi tương đối của sơ đồ và chiều dài xác định của nó. Như vậy, với sơ đồ trên trung bình có chiều dài xác định ngắn vẫn có thể có số chuỗi cá thể khớp với nó và tốc độ tăng theo lũy thừa. Với sơ đồ S_0 đang xét, ta có:

$eval(S,t)/\overline{F(t)}\left[1 - p_c\frac{\delta(S)}{m-1}\right]$ = 1.396751*0.984375=1.374927

Điều này có nghĩa là sơ đồ ngắn, trên trung bình S_0 vẫn nhận được số chuỗi tăng theo lũy thừa trong các thế hệ tiếp theo. Vào thời điểm (t+1) ta có 3*1.374927=4.12 chuỗi phù hợp với S_0(chỉ nhỏ hơn 4.19 là giá trị ta có khi tính trong chọn lọc). Vào thời điểm (t+2) ta có $3*1.374927^2$ =5.67 chuỗi như vậy (chỉ kém hơn 5.85 một chút).

2.3.3.Nguyên lý hoạt động của phép Đột biến

Phép đột biến thay đổi một vị trí trong nhiễn sắc thể một cách ngẫu nhiên với xác suất p_m, thay đổi từ 0 thành 1 và ngược lại. Như vậy, tất cả các vị trí cố định của sơ đồ phải giữ không đổi nếu sơ đồ muốn tồn tại qua đột biến.

Ví dụ: xét chuỗi v'_{19}

1	1	1	0	1	1	1	0	1	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v'_{19}

Và sơ đồ S_0

S_0=(****111**************************)

Giả sử chuỗi v'_{19} tham gia đột biến, nghĩa là có ít nhất một bit bị đảo. v'_{19} đột biến tại vị trí thứ 9 nên kết quả của nó là v''_{19} vẫn phù hợp với sơ đồ S_0

1	1	1	0	1	1	1	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	1	1	0	v''_{19}

Nếu các vị trí đột biến được chọn là từ 1 đến 4 hay từ 8 đến 33 thì chuỗi kết quả vẫn phù hợp với S_0, chỉ có 3 bít là 5, 6, 7 – các vị trí bít cố định trong S_0 là quan trọng. Nếu biến ít nhất một trong các bít này thì sơ đồ S_0 sẽ bị loại bỏ, số các bít quan trọng này bằng bậc của sơ đồ, nghĩa là bằng số các vị trí cố định.

Vì xác suất thay đổi của một bít là p_m nên xác suất tồn tại của một bít là $1-p_m$. Một lần đột biến độc lập với các đột biến khác , vì vậy xác suất tồn tại của sơ đồ S qua đột biến (nghĩa là chuỗi các đột biến một bit) là: $p_s(S) = (1 - p_m)^{o(S)}$ do $p_m<<1$ nên xác suất này có thể được tính gần đúng là $p_s(S) \approx 1 - o(S) * p_m$

Với sơ đồ S_0 và $p_m=0.01$ ta có $p_s(S) \approx 1 - 3*0.01 = 0.97$

Tác động kết hợp của chọn lọc, lai tạo và đột biến ta thấy dạng mới của phương trình tăng trưởng sơ đồ sinh sản là:

$$\xi(S, t+1) \geq \xi(S, t)*eval(S,t)/\overline{F(t)}\left[1 - p_c\frac{\delta(S)}{m-1} - o(S).p_m\right] \quad (2.3)$$

Tương tự như phương trình (2.1), (2.2), phương trình (2.3) cho ta biết kỳ vọng số chuỗi phù hợp với sơ đồ S trong thế hệ tiếp theo là hàm theo số chuỗi phù hợp với sơ đồ, thích nghi tương đối của sơ đồ, chiều dài xác định và bậc của sơ đồ. Ngoài ra, ta vẫn có kết quả là các sơ đồ trên trung bình có chiều dài xác định ngắn và bậc thấp vẫn có số chuỗi phù hợp với tốc độ tăng theo lũy thừa.

Đối với sơ đồ S_0

$$eval(S,t)/\overline{F(t)}\left[1 - p_c\frac{\delta(S)}{m-1} - o(S).p_m\right] = 1.396751*0.954375=1.333024$$

Điều này có nghĩa là sơ đồ ngắn, bậc thấp, trên trung bình S_0 vẫn nhận một số chuỗi tăng theo lũy thừa trong các thế hệ kế tiếp. Với t+1, ta có 3*1.333024=4.00 chuỗi phù hợp với S_0 (không kém 4.19 nhiều lắm- là giá trị tính ở bước chọn lọc hay 4.12 là giá trị tính được khi thực hiện chọn lọc và lai tạo). Với t+2, ta có $3*1.333024^2$ =5.33 chuỗi phù hợp với S_0 (không kém 5.85 hay 5.67 nhiều lắm). Lưu ý, phương trình (2.3) dựa trên giả thiết là hàm thích nghi f chỉ trả về các giá trị dương, khi áp dụng giải thuật GA vào những bài toán tối ưu có thể trả về các giá trị âm ta cần có một số ánh xạ bổ sung giữa các hàm tối ưu và độ thích nghi.

Chương 3
ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

3.1 Bài toán vận tải tuyến tính

Bài toán vận tải là một trong những bài toán tối ưu có ràng buộc đã được nghiên cứu chi tiết trong nhiều công trình. Mục đích là xây dựng một dự án vận tải với chi phí thấp nhất để vận chuyển một loại hàng hóa từ một số nguồn đến một số đích. Bài toán yêu cầu cho biết khả năng cung ứng tại từng nguồn, nhu cầu của mỗi đích và chi phí vận chuyển từ mỗi nguồn đến mỗi đích.

Do chỉ có một loại hàng hóa, mỗi đích có thể nhận hàng từ 1 hay nhiều nguồn. Mục tiêu là tìm số lượng cần chuyên chở từ mỗi nguồn đến mỗi đich sao cho tổng chi phí chuyên chở là tối thiểu.

Bài toán vận tải là tuyến tính nếu chi phí tỉ lệ với số lượng hàng vận tải nếu không nó là phi tuyến tính.

- Giả sử có n nguồn và k đích. Số cung tại nguồn i là S[i] và số cầu tại đích j là D[j], chi phí vận tải 1 đơn vị hàng giữa nguồn i và đích j là C[i,j]. Nếu x_{ij} là số lượng hàng hóa được vận tải từ nguồn i đến đích j thì bài toán vận tải đã cho là bài toán tối ưu.

$$\text{Min} \sum_{i=1}^{n} \sum_{j=1}^{k} fij(xij) \qquad \text{với} \begin{cases} \sum_{j=1}^{k} xij \le S[i]\ \forall i = \overline{1.n} \\ \sum_{i=1}^{n} xij \ge D[j]\ , \forall j = \overline{1.k} \end{cases}$$

  - Tập ràng buộc thứ nhất quy định lấy số lượng hàng được chỉ từ 1 nguồn không thể vượt quá số cung của nó.
  - Tập thứ 2 yêu cầu số lượng hàng trở đến đích phải thỏa số cầu của nó.
  - Nếu $f_{ij}(x_{ij})$ = C[i,j] * x_{ij} với mọi i, j thì bài toán là tuyến tính.
  - Bài toán trên bao hàm rằng tổng số cung $\sum_{i=1}^{n} S[i]$ ít nhất phải bằng với tổng số cầu $\sum_{j=}^{k} D[j]$. Khi tổng số cung bằng tổng số cầu, bài toán vận tải gọi là bài toán vận tải cân bằng. Bài toán vận tải cân bằng khác bài toán vận tải tuyến tính ở chỗ tất cả các ràng buộc tương ứng là các phương trình, nghĩa là $\sum_{j=1}^{k} x_{ij}$

$$\sum_{j=1}^{k} x_{ij} = S[i]\ ,\ i=1, 2, 3\ldots, n$$

$$\sum_{i=1}^{n} x_{ij} = D[j]\ ,\ j=1, 2, 3\ldots, k$$

Nếu tất cả các S[i] và D[j] là các số nguyên, một lời giải tối ưu bất kỳ của bài toán vận tải tuyến tính cân bằng là 1 lời giải nguyên nghĩa là tất cả các x_{ij} (i=1, 2…n ; j=1, 2…k) là những số nguyên. Số lượng các số nguyên dương x_{ij} tối đa là (k+n-1).

Ví dụ : Xét bài toán vận tải có 3 nguồn và 4 đích, số cung tương ứng là: S(1) =15; S(2) = 25; S(3) = 5

- Số cầu tương ứng là:D[1] = 5; D[2] = 15; D[3] = 15; D[4] = 10

$\sum_{i=1}^{3} S[i] = 45 = \sum_{j=1}^{4} D[j]$ Tổng cung = tổng cầu.

- Chi phí vận chuyển 1 đơn vị C[i,j] với i=1, 2, 3; j =1, 2, 3, 4 được cho trong bảng sau:

$$C_{ij} = \begin{pmatrix} 10 & 0 & 20 & 11 \\ 12 & 7 & 9 & 20 \\ 0 & 14 & 16 & 18 \end{pmatrix}$$

Lời giải tối ưu được cho trong bảng sau, tổng chi phí là 315, lời giải gồm các giá trị nguyên của x_{ij}

Số lượng vận tải:

	5	15	15	10
15	0	5	0	10
25	0	10	15	0
3	5	0	0	0

Giải thuật di truyền giải bài toán vận tải cân bằng.

Một lời giải (1 nhiễm sắc thể) là một vecto bít $<v_1, v_2, \dots v_p>$, (p=n*k) sao cho mỗi thành phần v_i (i=1, 2, …p) là 1 vecto bit $< w_0^i,\dots,w_s^i>$ và biểu diễn 1 số nguyên liên quan đến hàng j và cột m trong ma trận phân phối, ở đây j = $[\frac{(i-1)}{k} + 1]$ và m= (i-1) mod k+1.

Chiều dài của vecto w (tham số của f) quyết định cố nguyên lớn nhất (2^{s+1}-1) cần có để biểu diễn một lời giải.

a. Thỏa mãn ràng buộc: rõ ràng là mỗi vecto lời giải phải thỏa mãn các ràng buộc sau đây:

$$\sum_{i=ck+1}^{ck+k} Vi = S[c+1]; \qquad với\ c = 0,1,2,3\dots,n-1$$

$$\sum_{j=m,stepk}^{kn} Vj = D[m]; \quad với\, m = 0,1,2,3 \ldots, k$$

$$Vq \geq 0 \quad với\, q = 1,2,3 \ldots n; j = 1,2 \ldots . k - n$$

Ràng buộc cuối cùng luôn được thỏa mãn (1 chuỗi các số 0, 1 được đổi ra là 1 số nguyên dương). Hai ràng buộc đầu thể hiện tổng cung bằng tổng cầu tại mỗi nguồn và đích (mặc dù các công thức này là không đối xứng).

b. Hàm lượng giá:

Hàm lượng giá đơn giản là tính tổng chi phí vận tải từ các nguồn đến các đích và được tính theo:

Eval(<$v_1, v_2, \ldots v_n$>) = $\sum_{i=1}^{p} Vi * C[j,m]$

Trong đó j = $[\frac{(i-1)}{k} + 1]$ và m= (i-1) mod k +1.

- Các toán tử di truyền:

Đột biến được định nghĩa là thay đổi của 1 bit trong một vecto lời giải, điều này tương ứng với thay đổi của một giá trị nguyên v_i, đối với bài toán vận tải, đột biến này có thể gây ra một chuỗi những thay đổi ở những điểm khác nhau để duy trì các ràng buộc, ta cần ghi nhớ thay đổi đã xảy ra trong cột nào và hàng nào (các nguồn và các đích)=> điều này làm cho công thức trở nên phức tạp (mất tính đối xứng khi diễn tả các ràng buộc).

Giả sử 2 điểm ngẫu nhiên (v_i và v_m với i<m) được chọn sao cho chúng thuộc cùng một hàng hay cột.

Giả sử v_i, v_j , v_k, v_m (với i<j<k<m) là những thành phần của vecto lời giải (được chọn để đột biến) sao cho v_j và v_j cũng như v_k, $v_m$thuộc cùng 1 hàng.

Tức là trong biểu diễn ma trận ta có:

…. ……………

…. v_i…….v_j…..

…. ……………

…. v_k……v_m…

Với cách phân tích này thì gặp phải khó khăn khi xác định sự thay đổi cần thiết trong vecto lời giải, ta nên tăng hay giảm v_i?

Ta có thể chọn cách đổi nó thành một hay một số ngẫu nhiên nào để trong khoảng <0, 1,…,v_i>. Nếu tăng giá trị v_i theo một hằng số C ta phải giảm mỗi giá trị v_j và v_k theo cùng hằng số đó. Điều gì sẽ xảy ra nếu v_j<C hoặc v_k <C. Ta có

thể chọn C=min(v_i , v_j , v_k), với cách này phần lớn các đột biến sẽ có kết quả không thay đổi gì cả, do xác suất của việc chọn 3 phần tử không là có thể gần bằng 0 (nhỏ hơn 1/n đối với các vecto có kích thước n^2).

Như vậy, phương pháp thay đổi 1 bít làm cho các toán tử đột biến không hiệu quả vì cần những biểu thức phức tạp để kiểm soát hàng hay cột tương ứng của phần tử được chọn => biểu diễn vecto nhị phân như trên không phải là phương pháp thích hợp nhất để định nghĩa các toán tử di truyền trong những bài toán ràng buộc khác này. Người ta tìm cách để biểu diễn lời giải mà vẫn bảo tồn cấu trúc cơ bản của biểu diễn vecto khi thực hiện các phép di truyền. Việc này cần có sự kết hợp tri thức của bài toán vào biểu diễn này.

Ta xét một lời giải thỏa tất cả các ràng buộc, tên của thủ tục là *Khởi tạo*. Đây là thành phần cơ bản của toán tử đột biến trong các toán tử di truyền của cấu trúc 2 chiều, nó tạo ra 1 ma trận có tối đa (k+n-1) phần tử $\neq$0 sao cho tất các ràng buộc được thỏa mãn:

Thủ tục KHỞI TẠO

+ Input: Mảng D[k], S[n];

+Output: Mảng $(v)_{ij}$ sao cho $v_{ij} \geq 0$ với $\forall$i,j

Thỏa mãn

$$\sum_{j=1}^{k} Vij = D[i] \text{ với } i = 1,2 \ldots m$$

$$\sum_{i=1}^{n} Vij = S[j] \text{ với } j = 1,2 \ldots k$$

Procedure Khoitao;

L <- {1, 2, …, k*n}: là danh sách các điểm chưa được xét đến;

Lặp

1. Chọn 1 số ngẫu nhiên q trong L;
2. Q được gọi là điểm đã xét.
3. i := $[\frac{(q-1)}{k} + 1]$
4. j := ((q-1) mod k +1).
5. Val := min (S[i], D[j])
6. V_{ij} := Val ;
7. S[i] := S[i] – Val

8. D[i] := D[i] – Val

Hết lặp nếu (tất cả các điểm trong L đều đã được thăm);

End;

Xét bài toán vận tải 3 nguồn, 4 đích, các tham số:

S[1] = 15; S[2] = 25 ; S[3] = 5 ; i=1, 2, 3 ; n=3

D[1] = 5 ; D[2] = 15 ; D[3] = 15 ; D[4] = 10; j=1, 2, 3, 4 ; k=4

Có tất cả 3*4 = 12 số đều chưa được thăm từ đầu. Chọn số ngẫu nhiên đầu tiên chẳng hạn là 10 với số này tính được.

- Hàng i = 3 và cột j = 2 (q=10; i = $[\frac{(10-1)}{4} + 1] = 3$;

  j= (10-1) mod 4 +1 = 2)
- Val = min (S[3], D[3]) = min (5, 15) = 5
- $V_{ij} = V_{32} = 5$; (S[3] = S[3] – 5= 5-5 = 0) ; S=(15, 25, 0)

  D[2] = D[2] – 5= 15-5 = 10 ; D = (5, 10, 15, 10).

Ta lặp lại cách tính này với 3 số (chưa thăm) ngẫu nhiên, ví như 8, 5 và 3.

+ Với số 8 : i= $[\frac{(8-1)}{4} + 1] = 2$

j= (8-1) mod 4 +1 = 4.

Val = min (S[2], D[4]) = min (25, 10) = 10

$V_{ij} = V_{24} = 10$; (S[2] = S[2] – 10= 25 - 10 = 15) ; S=(15, 15, 0)

D[4] = D[4] – 10= 10-10 = 0 ; D = (5, 10, 15, 0).

+ Với số 5 : i= $[\frac{(5-1)}{4} + 1] = 2$

j= (5-1) mod 4 +1 = 4.

Val = min (S[2], D[1]) = min (15, 5) = 5

$V_{ij} = V_{21} = 5$; (S[2] = S[2] – 5= 15 - 5 = 10) ; S=(15, 10, 0)

D[1] = D[1] – 5 = 5 - 5 = 0 ; D = (0, 10, 15, 0).

+ Với số 3 : i= $[\frac{(3-1)}{4} + 1] = 1$

j= (3 - 1) mod 4 + 1 = 3.

Val = min (S[1], D[3]) = min (15, 15) = 15

$V_{ij} = V_{13} = 15$; (S[1] = S[1] – 15= 15 - 15 = 0) ; S=(0, 10, 0)

D[3] = D[3] – 15 = 15 - 15 = 0 ; D = (0, 10, 0, 0).

Khi đó ta có bảng kết quả sau:

	0	10	0	0
0			15	

10	5			10
0		5		

Các giá trị S[i] và D[i] là những số có được sau 4 lần lặp. Nếu có thêm các số ngẫu nhiên là 1, 11, 4, 12, 7, 6, 9, 2 thì tính được ma trận cuối cùng. (với chuỗi các số ngẫu nhiên được chấp nhận (10, 8, 5, 3,1, 11, 4, 12, 7, 6, 9, 2) là

	0	0	0	0
0	0	0	15	0
0	5	10	0	10
0	0	5	0	0

Như vậy, sau 12 lần lặp (các bản sao cục bộ) thì S[i] và D[j] = 0. Lưu ý, có nhiều chuỗi số mà thủ tục khởi tạo có thể tạo ra được lời giải tối ưu cho chúng. Chẳng hạn, lời gải tối ưu (trong ví dụ trên) có thể nhận được đối với bất cứ chuỗi nào sau đây: <7, 9, 4, 2, 6, *, *, *, *, *, *, *> (* là số chưa thăm bất kỳ) cũng như đối với nhiều chuỗi khác .

Kỹ thuật này có thể phát sinh 1 lời giải khả thi chứa tối đa (k+n-1) phần tử nguyên $\neq 0$. Nó sẽ không phát sinh những lời giải khác, có thể cũng khả thi, nhưng không dùng chung đặc trưng này (Thủ tục khởi tạo sẽ phải được hiệu chỉnh khi giải bài toán vận tải phi tuyến).

Dựa trên tri thức và những lời giải của bài toán cho ta một cách khác để biểu diễn lời giải của bài toán vận tải bằng 1 vecto. Vecto lời giải là 1 chuỗi các số nguyên phân biệt k*n trong khoảng <1, k*n>, theo thủ tục khởi tạo sẽ sinh ra 1 lời giải khả thi. Nói cách khác, ta sẽ xem vecto lời giải là 1 hoán vị của các số và ta có thể tìm các hoán vị cụ thể tương ứng với lời giải tối ưu.

Tiếp theo ta sẽ xét đến cách cài đặt của biểu diễn này trong việc thỏa mãn các ràng buộc, hàm lượng giá và các toán tử di truyền.

1.Thỏa mãn ràng buộc:

Bất cứ hoán vị của k*n số phân biệt sẽ sinh ra một lời giải duy nhất thỏa mãn tất cả các ràng buộc, điều này được đảm bảo bởi thủ tục khởi tạo.

2.Hàm lượng giá

Hoán vị nào cũng tương ứng với một ma trận duy nhất như (v_{ij}), hàm lượng giá là.

$$\sum_{i=1}^{n}\sum_{j=1}^{k} Vij * C[i,j]$$

3.Các toán tử di truyền

3.1 Đảo

Nếu <$x_1, x_2, \ldots, x_q$> (q=k*n) là một lời giải thì vecto đảo của nó <$x_q, x_{q-1}, \ldots, x_1$> cũng là một lời giải.

3.2 Đột biến: Bất cứ 2 phần tử nào của vecto lời giải <$x_1, x_2, \ldots, x_q$>, x_i và x_j chẳng hạn cũng có thể được hoán vị và vecto kết quả cũng là một lời giải.

3.3 Lai tạo (phức tạp hơn 1 chút)

Một toán tử lai ngẫu nhiên có sinh ra các lời giải không hợp lệ (Ví dụ: Áp dụng toán tử lai như vậy vào các chuỗi).

*<1, 2, 3, 4, 5, 6,/ 7, 8, 9, 10, 11, 12> và <7, 3, 1, 11, 4, 12, /5, 2, 10, 9, 6, 8> Sẽ sinh ra (điểm lai tạo là ở sau vị trí thứ 6)

Các lời giải <1, 2, 3, 4, 5, 6, 5, 2, 10, 9, 6, 8> và <7, 3, 1, 11, 4, 12, 7, 8, 9, 10, 11, 12> cả 2 chuỗi này đều là các lời giải không hợp lệ.

Như vậy, phải dùng tới một toán tử lai Heuristie mà khi cho 1 cặp cha mẹ, sẽ tạo một con hợp lệ theo thủ tục sau:

1. Tạo bản sao của phần tử thứ hai trong cặp cha – mẹ đã cho.
2. Chọn một phần ngẫu nhiên từ phần tử thứ nhất trong cặp cha – mẹ đó.
3. Tạo những thay đổi tối thiểu cần thiết ở con để có được lời giải hợp lệ.

Ví dụ: Với cặp cha – mẹ (*) đã xét ở ví dụ trên, phần được chọn là (4, 5, 6, 7) thì con có được là <3, 1, 11, 4, 5, 6, 7, 12, 2, 10, 9, 8>.

Với con này ta thấy nó cỏ mỗi liên hệ cấu trúc với cả cha và mẹ. Rồi vai trò của cặp cha mẹ này có thể được thay đổi khi sinh con thứ 2.

Biểu diễn lời giải bài toán vận tải bằng ma trận:

Biểu diễn lời giải tự nhiên nhất cho bài toán vận tải là cấu trúc 2 chiều. Hay ma trận V= (v_{ij}) với i = 1, 2, …n; j= 1, 2, ..,k có thể biểu diễn lời giải.

1. Thỏa mãn ràng buộc

   Rõ ràng mọi ma trận lời giải V = (v_{ij}) có thể thỏa mãn các ràng buộc

$$\sum_{j=1}^{k} Vij = S[i] \text{ với } i = 1,2 \ldots n$$

$$\sum_{i=1}^{n} Vij = D[j] \ \text{với}\ j = 1,2 \ldots k$$

2. Hàm lượng giá: Là hàm mục tiêu thông thường

   Eval(<v_{ij}) = $\sum_{i=1}^{n}\sum_{i=1}^{p} Vij * C[j,m]$

   Trong đó j = $[\frac{(i-1)}{k} + 1]$ và m= (i-1) mod k +1

Công thức này đơn giản hơn nhiều so với PP dể hiểu và nhanh hơn trong hệ GENETIC – I mà mỗi chuỗi phải được biến đổi (khởi tạo) thành 1 ma trận lời giải trước khi lượng giá.

3. Các toán tử:

   Ta định nghĩa 2 toán tử di truyền là đột biến và lai (thật khó có thể định nghĩa một toán tử đảo có ý nghĩa trong trường hợp này).

   3.1 Đột biến

- Giả sử rằng {i_1, i_2, …i_p} là tập con của {1, 2, 3…, n} và {j_1, j_2, …j_q} là tập con của {1, 2, ….,k} sao cho 2$\le p \le n; 2 \le q \le k$.
- Ta biểu diễn một cha (mẹ) của đột biến bằng ma trận (n*k) V= (v_{ij}) rồi tạo ra một ma trận con (p x q), W=(w_{ij}) từ tất cả các phần tử của ma trận V theo cách sau: một phần tử v_{ij}∈ V ở trong W ⇔ i ∈ {i_1, i_2 ,….i_p} và j ∈ {j_1, j_2 ,….j_q} (nếu i = i_r và j = j_s thì phần tử v_{ij} được đặt trong hàng r và cột s của ma trận W).
- Sau đó ta gán các giá trị mới S W[i] và D W[j] (1$\le i \le p; 1 \le j \le q$) cho ma trận W:

$$S\ \ W[i] = \sum_{j \in \{j1, j2 \ldots, jq\}} Vij \ \text{ với } 1 \le i \le p$$

$$D\ \ W[j] = \sum_{i \in \{i1, i2 \ldots, ip\}} Vij \ \text{ với } 1 \le j \le q$$

Có thể dùng thủ tục khởi tạo để gán các giá trị mới vào ma trận W sao cho tất cả các ràng buộc S W[i] và D W[i] được thỏa mãn. Sau đó, thay những phần tử thích hợp của ma trận V bằng các phần tử mới của ma trận W. Với cách này, tất cả các ràng buộc toàn cục (S[i] và D[j]) được bảo toàn.

Xét ví dụ sau minh họa cho toán tử đột biến:

Ví dụ :Xét bài toán vận tải có 4 nguồn (n=4) và 5 đích (k=5) với những ràng buộc : S [1] = 8 ; S [2] = 4 ; S [3] = 12 ; S [4] = 6 ; S = (8,4, 12, 6)

D [1] = 3 ; D [2] = 5 ; D [3] = 10 ; D [4] = 7; D [5] = 5; D=(3,5,10,7,5).

giả sử ma trận V sau đây được chọn làm cha (mẹ) cho đột biến

V   W

0	0	5	0	3
0	4	0	0	0
0	0	5	7	0
3	1	0	0	2

4	0	0
1	0	2

Chọn ngẫu nhiên 2 hàng l à 2,4 và 3 cột : 2, 3, 5.Khi đó có ma trận con W tương ứng.

Ta có : S W [1] = 4 (= $v_{22} + v_{23} + v_{25} = 4 + 0 + 0 = 4$)

S W [2] = 3 (= $v_{42} + v_{43} + v_{45} = 1 + 0 + 2 = 3$)

D W [1] = 5 (= $v_{22} + v_{42} = 4 + 1 = 5$)

D W [2] = 0 (= $v_{23} + v_{43} = 0 + 0 = 0$)

D W [3] = 2 (= $v_{25} + v_{45} = 0 + 2 = 2$)

Sau khởi tạo lại ma trận W, ma trận có những giá trị sau:

2	0	2
1	0	2

Cuối cùng con của ma trận V sau đột biến là:

0	0	5	0	3
	2	0	0	2
0	0	5	7	0
3	3	0	0	0
3	3	0	0	2

3.2 Lai tạo : giả sử rằng hai ma trận V1= (v^1_{ij}) và V2= (v^2_{ij}) đ ư ợc chọn làm cha (mẹ) đ ể thực hiện lai .Ta xét giải thuật được dùng để sinh 2 con V_3,V_4.

+ T ạo 2 ma trận tạm: DIV =(d_iv_{ij}) và REM = (rem_{ij}) với

$div_{ij} = [(v^1_{ij} + v^2_{ij})/2]$

$rem_{ij} = [(v^1_{ij} + v^2_{ij}) \bmod 2]$

Như vậy:

+ Ma trận DIV lưu các giá trị trung bình được làm tròn từ cha mẹ.

+ Ma trận REM theo dõi việc làm tròn nào là cần thiết.

Ma trận REM có một số tính chất:

- Số các số 1 trong mỗi hàng và mỗi cột là bằng nhau. Hay các giá trị của

S REM[i] và D REM[j] là những số nguyên chẵn. Ta dung tính chất này để biến đổi REM thành 2 ma trận REM1 & REM2 sao cho:

REM = REM1 + REM2

S REM1[i] = S REM2[i] = S REM[i]/2 với i = 1, 2, ..., n

D REM1[j] =D REM2[j] = D REM[j]/2 với j = 1, 2, ..., k

Sau đó sinh ra 2 con của V là V_1 , V_2.

V_3 = DIV + REM1;

V_4 = DIV + REM2.

Ví dụ: Minh hoạ phép lai.

Giả sử các ma trận V_1 và V_2 sau đây đã được chọn làm cha mẹ để lai tạo

$$V_1 = u = \begin{pmatrix} 1 & 0 & 0 & 7 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 2 & 1 & 4 & 0 & 5 \\ 0 & 0 & 6 & 0 & 0 \end{pmatrix} \quad ; \quad V_2 = V = \begin{pmatrix} 0 & 0 & 5 & 0 & 3 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 5 & 7 & 0 \\ 3 & 1 & 0 & 0 & 2 \end{pmatrix}$$

Các ma trận DIV và REM là:

$$DIV = \begin{pmatrix} 0 & 0 & 2 & 3 & 1 \\ 0 & 4 & 0 & 0 & 0 \\ 1 & 0 & 4 & 3 & 2 \\ 1 & 0 & 3 & 0 & 1 \end{pmatrix} \quad ; \quad REM = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Hai ma trận REM1 & REM2 là:

$$REM\ 1 = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$REM\ 2 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Hai ma trận con là: C_1 = DIV + REM1; C_2 = DIV + REM2

$$C_1 = \begin{pmatrix} 0 & 0 & 3 & 3 & 2 \\ 0 & 4 & 0 & 0 & 0 \\ 1 & 1 & 4 & 4 & 2 \\ 2 & 0 & 3 & 0 & 1 \end{pmatrix} \quad ; \quad C_2 = \begin{pmatrix} 1 & 0 & 2 & 4 & 1 \\ 0 & 4 & 0 & 0 & 0 \\ 1 & 0 & 5 & 3 & 3 \\ 1 & 1 & 3 & 0 & 1 \end{pmatrix}$$

3.2. Giải thuật di truyền và bài toán chiếc ba lô

3.2.1.Bài toán ba lô 0/1: Cho trước một tập thực thể cùng các giá trị và kích thước của chúng. Cần chọn được một hay nhiều tập con rời nhau sao cho tổng kích thước trong mỗi tập con không vượt quá giới hạn cho phép và tổng giá trị đã chọn là lớn nhất.

Đây là bài toán NP-khó nên có thể tiếp cận bằng cách dùng Heuristic. Giả sử với tập trọng số W[i]; giá trị P[i] và sức chứa của ba lô là C.

Ta xác định một véc tơ nhị phân x = <x[1], x[2], …, x[n]> sao cho:

$$\sum_{i=1}^{n} x_i w_i \le C \text{ và } P(x) = \sum_{i=1}^{n} x_i P_i \to \max$$

Khó khăn lớn của những bài toán này là chịu ảnh hưởng lớn bởi tương quan giữa lợi ích (giá trị của các đồ vật) và các trọng số (thể tích/ trọng lượn của các đồ vật), chúng ta khảo sát 3 tập dữ liệu thử:

1. Không tương quan W[i]:= Random([1 .. v]); P[i]:= Random ([1 .. v]);
2. Tương quan yếu W[i]:= Random([1 .. v]); P[i]:= W[i] +Random ([-r .. r]); (Nếu với i bất kỳ, P[i] ≤0 thì giá trị đó bị bỏ qua và phép tính được lặp lại cho tới khi P[i] >0)
3. Tương quan mạnh W[i]:= Random([1 .. v]); P[i]:= W[i] + r; Tương quan cao hơn có nghĩa là giá trị khác biệt nhỏ hơn.

$$Max_{i=1..n}\left\{\frac{P[i]}{W[i]}\right\} - Min_{i=1..n}\left\{\frac{P[i]}{W[i]}\right\}$$

(Những bài toán tương quan cao hơn có mức khó khăn cao hơn.)

Dữ liệu được phát sinh v = 10, r = 5; các thử nghiệm với 3 tập dữ liệu ứng với mỗi thực thể có n = 100; 250; 500 đồ vật. Với ba lô, ta quan tâm và xét đến 2 loại:

①Ba lô có sức chứa $C_1 = 2v$. Với trường hợp này, lời giải tối ưu chứa rất ít đồ vật. (Với những điều kiện không đầy đủ một vùng hầu như phụ trách toàn bộ miền).

②Ba lô có sức chứa $C_2 = 0.5 \sum_{i=1}^{n} W[i]$ (sức chứa trung bình). Với trường hợp này khoảng phân nửa các đồ vật nằm trong lời giải tối ưu. (Việc tăng sức chứa C không làm tăng nhiều thời gian tính toán của các giải thuật cổ điển).

1. Mô tả giải thuật: Ba giải thuật được đề cập, cài đặt và thử nghiệm:
   - Giải thuật dựa trên hàm thưởng phạt: Ap[i] với i là chỉ số của một giải thuật cụ thể trong lớp giải thuật này.

- Giải thuật dựa trên phương pháp sửa chữa Ar[i].
- Giải thuật dựa trên phương pháp giải mã Ad[i].

3.2.2.Một số GA giải bài toán chiếc ba lô

a. **Giải thuật Ap[i]:** Dựa trên hàm thưởng phạt.

- Lời giải của bài toán được biểu diễn bởi một chuỗi nhị phân có độ dài là n.

x = (x[1], x[2],…, x[n]), đồ vật thứ i được chọn và đặt vào ba lô nếu và chỉ nếu x[i] = 1.

- Độ thích nghi eval(x) của mỗi chuỗi được định nghĩa là:

$$\text{eval(x)} = \sum_{i=1}^{n} x[i]P[i] - Pen(x)$$

Trong đó: Pen(x) = 0 với lời giải khả thi x, nghĩa là các lời giải thỏa mãn $\sum_{i=1}^{n} x_i w_i \le C$; Pen(x) > 0 trong trường hợp ngược lại.

Có nhiều chiến lược để gán giá trị thưởng phạt, ta xét đến 3 trường hợp mà hàm thưởng phạt tăng theo logarit, tuyến tính và bình phương tùy theo mức độ xâm phạm.

①Ap[1]: $Pen(x) = \log_2[1 + \int(\sum_{i=1}^{n} x_i w_i - C)]$

②Ap[2]: $Pen(x) = \rho(\sum_{i=1}^{n} x_i w_i - C)$

③Ap[3]: $Pen(x) = (\rho(\sum_{i=1}^{n} x_i w_i - C))^2$

Trong cả 3 trường hợp lấy $\rho = \max_{i=1..n}\left\{\frac{P[i]}{W[i]}\right\}$

b. **Giải thuật Ar[i]:**

- Lời giải của bài toán được biểu diễn bởi một chuỗi nhị phân có độ dài n.

x =(x[1], x[2],…, x[n]), đồ vật thứ i được chọn và đặt vào ba lô nếu và chỉ nếu x[i] =1.

- Độ thích nghi của mỗi chuỗi được định nghĩa là:

$\text{eval(x)} = \sum_{i=1}^{n} x'[i]Pi$, trong đó x’i là phiên bản được sửa chữa của véc tơ gốc x.

Với giải thuật này ta nhận thấy:

- Có thể có nhiều phương pháp sửa chữa khác nhau.

- Có vài % các nhiễm sắc thể được sửa chữa có thể thay cho các nhiễm sắc thể gốc trong quần thể, tỉ lệ thay thế đó thay đổi từ 0% tới 1% hoặc theo luật 5% (luật này cho rằng, trong thực nghiệm, nếu thay các nhiễm sắc thể gốc với xác suất 5% thì hiệu quả của giải thuật sẽ tốt hơn khi thay thế bằng tỷ lệ khác.

Xét thủ tục sửa chữa (x):

Procedure sửachữa (x);

{OK = true nếu ba lô đầy và ngược lại}

Begin

1. OK:= false;

x' := x;

2. if $\sum_{i=1}^{n} x_i' w_i > C$ then OK:= true

3. While OK do

Begin

+ i:= chọn một đồ vật trong ba lô, loại đồ vật này ra khỏi ba lô, nghĩa là x'[i]=0;

+ if $\sum_{i=1}^{n} x_i' w_i \le C$ then OK:= false;

End;

End;

Xuất phát từ thủ tục sửa chữa trên, ta có 2 giải thuật sửa chữa với sự khác nhau nằm ở câu lệnh chọn của thủ tục chọn lọc, lệnh này chọn một đồ vậy để lấy nó ra khỏi ba lô.

+ Ar[1] (sửa chữa ngẫu nhiên): Thủ tục chọn lọc chọn ngẫu nhiên một phần tử trong ba lô.

+ Ar[2] (sửa chữa tham lam): Tất cả các đồ vật trong ba lô được sắp theo thứ tự giảm dần về lợi ích của chúng đối với tỉ lệ trọng số, thủ tục chọn lọc luôn luôn chọn đồ vật cuối cùng (trong danh sách các đồ vật có sẵn) để xóa bỏ.

c. **Giải thuật Ad[i]:** một bộ giải mã cho bài toán ba lô được biểu diễn nguyên (Integer).

Trong giải thuật này dùng biểu diễn thứ tự các đồ vật được chọn.

+ Mỗi nhiễm sắc thể là một véc tơ có n số nguyên, thành phần thứ i của véc tơ là một số nguyên trong khoảng từ 1 đến (n-i +1).

+ Biểu diễn thứ tự tham chiếu đến danh sách L các đồ vật. Một véc tơ được giải mã bằng cách chọn đồ vật thích hợp trong danh sách hiện hành.

Ví dụ: với danh sách L (1, 2, 3, 4, 5, 6).

Véc tơ <4, 3, 4, 1, 1, 1> được giải mã thành các đồ vật được chọn là: (4, 3, 6, 1, 2, 5). L = (1, 2, 3, 4, 5, 6)

Thành phần thứ nhất trong vecto là 4 $\Rightarrow$ chọn đồ vật 4. Loại 4 ra khỏi danh sách. Đồ vật thứ 3 lúc này là 3 $\Rightarrow$ loại 3 ra khỏi danh sách. Đồ vật thứ 4 là 6 $\Rightarrow$loại 6 ra khỏi danh sách. Đồ vật thứ 1 là 1 $\Rightarrow$ loại 1 ra khỏi danh sách. Đồ vật thứ 1 là 2 $\Rightarrow$ loại 2 ra khỏi danh sách. Đồ vật thứ 1 là 5, loại 5 ra khỏi danh sách.

Với giải thuật này, một nhiễm sắc thể có thể được hiểu là chiến lược kết hợp các đồ vật vào lời giải.

- Lai một điểm được áp dụng cho hai cha – mẹ hợp lệ bất kỳ cũng sẽ cho ra một con hợp lệ.
- Toán tử đột biến được định nghĩa theo cùng cách với biểu diễn nhị phân: Nếu gen thứ i bị đột biến, nó sẽ có một giá trị ngẫu nhiên trong khoảng [1..n-i+1].
- Ta xét thủ tục giải mã (x):

```
Procedure giải mã (x);
        Begin
          1. Xây dựng danh sách L;
             i:=1;
             tổng trọng:=0;
             Tổng lợi ích :=0;
          2. While i <=n do
                Begin
                + j := x[i];
                + Loại bỏ đồ vật thứ j ra khỏi danh sách L;
                + if Tổng trọng + W[j] ≤ C then
                     Begin
                     Tổng trọng := tổng trọng + W[j];
                     Tổng lợi ích := tổng lợi ích + P[j];
                     End;
                + i:= i+1;
```

End;

End;

Với thủ tục giải mã (x) có 2 giải thuật dựa trên sự khác nhau ở thủ tục xây dựng.

1. Ad[1](giải mã ngẫu nhiên): Tạo danh sách L sao cho thứ tự của đồ vật trong danh sách tương ứng với các đồ vật file nhập (ngẫu nhiên).

- Ad[2] (giải mã tham lam): tạo danh sách L theo thứ tự giảm lợi ích theo tỷ lệ trọng số. Việc giải mã mà véc tơ x được thực hiện trên cơ sở thứ tự sắp xếp (có một số điểm tương tự với Ar[2]).

Ví dụ: x[i] = 23 được hiểu là đồ vật thứ 23 theo thứ tự giảm lợi ích đối với tỷ lệ trọng số trên danh sách hiện hành L.

Nhận xét:

- Các hàm thưởng phạt Ap[i] ($\forall$i) không cho kết quả tốt với những bài toán có sức chứa ba lô nhỏ (C_1).
- Với các bài toán (C_2), sức chứa của ba lô trung bình, giải thuật Ap[1] dựa trên hàm thưởng phạt logarit khá thành công, nó hiệu quả hơn các phương pháp khác qua mọi trường hợp, nhưng nó bị hạn chế ở những bài toán có sức chứa hạn chế.
- Phương pháp A_r[2] (sửa chữa tham lam) có hiệu quả hơn các phương pháp khác trong trường hợp C_1

2. Giải thuật sửa chữa có hiệu quả tốt trong trường hợp C_2 (nếu chứa trung bình).
3. Có thể kết hợp các phương pháp thưởng phạt với sửa chữa để những lời giải không khả thi của giải thuật Ap[i] có thể được sửa chữa thành khả thi.
4. Với giải thuật giải mã cần phải thử nghiệm với nhiều biểu diễn khác nhau: biểu diễn kề (lai với các cạnh thay đổi, đột biến các phần của hành trình con, hay đột biến Heuristic) hoặc biểu diễn đường dẫn (với các đột biến PMX, OX, CX hay đột biến tái kết hợp).

3.3 Bài toán tối ưu số

3.4.1. Giới thiệu chung:

Khi ứng dụng GA vào thực tế ta thường gặp những bài toán yêu cầu một cách biểu diễn lời giải thích hợp, nếu không GA khó có thể cho lời giải tốt. Có thể là do GA thường hội tụ sớm về một lời giải tối ưu không toàn cục, cũng có thể là do gặp bài toán tối ưu với các ràng buộc không tầm thường.

Biểu diễn nhị phân truyền thống có một số bất lợi khi áp dụng GA giải các bài toán số cần độ chính xác cao, trong một không gian số chiều lớn, khi đó GA thực hiện với hiệu quả kém không cao. Cách biểu diễn nhị phân khiến cho bât kỳ cách mã hóa nào cũng có cùng số sơ đồ, chính vì vậy, cách dùng chuỗi bit để mã hóa lời giải là một trong những công việc chính của những người nghiên cứu GA.. Cách mã hóa nhị phân thường giúp dễ dàng cho việc phân tích lý thuyết và cho phép xây dựng các toán tử di truyền 'đẹp', nhưng kết quả thực sự không phụ thuộc vào việc sử dụng các chuỗi bit mà còn thử nghiệm với các tập mẫu lớn (không phải chỉ có 0 và 1) và các toán tử di truyền mới.

Đặc biệt với các bài toán tối ưu số với biến liên tục ta có thể thử nghiệm với các gen mã hóa là các số thực cùng với các toán tử di truyền đặc biệt. Với phương pháp này các gen sẽ nhận giá trị thực. Với mục đích chính là mở rộng không gian tìm kiếm của GA đến gần không gian thực của bài toán, bởi vậy các phép toán cũng được sửa đổi bằng cách sử dụng một số đặc trưng cụ thể của không gian thực.

Ví dụ biểu diễn sẽ xét có thuộc tính là hai điểm gần nhau trong không gian biểu diễn cũng phải gần nhau trong không gian của bài toán và ngược lại. Điều này không phải lúc nào cũng đúng trong biểu diễn nhị phân. Tuy nhiên, nếu dùng cách biểu diễn nhị phân nhưng theo nguyên tắc Gray cũng có thể mở rộng không gian tìm kiếm của GA gần với không gian thực của bài toán.

Thủ tục để chuyển một số nhị phân $b=(b_1, b_2, ...,b_m)$ thành số mã Gray $g=(g_1, g_2, ...,g_m)$ và ngược lại: (m là số bit trong các biểu diễn này)

Procdure Binary_to_Gray;

Begin

1. $g_1=p_1$
2. For k=2 to m do $g_k:= b_{k-1}$ XOR b_k

End;

Procdure Gray_to_binary;

Begin

1. gia_tri:=g_1; b_1:=giatri;
2. For k=2 to m do

```
        begin
          If  gk= 1 then gia_tri:=NOT gia_tri
          bk:=gia_tri;
         end;
End;
```

Bảng mã 16 số nhị phân cùng các mã Gray tương ứng:

Binary	Gray
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

Trong biểu diễn Gray có đặc điểm là hai điểm gần nhau bất kỳ trong không gian bài toán chỉ khác nhau một bit. Hay việc tăng giá trị tham số lên một tương ứng với việc thay đổi chỉ một bit trong mã. Có khá nhiều phương pháp để chuyển đổi giữa nhị phân và Gray. Xét ví dụ với m=4, cặp ma trận:

$$A=\begin{bmatrix}1&0&0&0\\1&1&0&0\\0&1&1&0\\0&0&1&1\end{bmatrix} \qquad A^{-1}=\begin{bmatrix}1&0&0&0\\1&1&0&0\\1&1&1&0\\1&1&1&1\end{bmatrix}$$

Cho ra hai biến đổi sau: g=Ab và $b=A^{-1}g$

Tuy nhiên, ta có thể dùng biểu diễn dấu chấm động thay vì biểu diễn nhị phân dạng Gray, vì nó gần với không gian bài toán hơn và cũng cho phép dễ cài đặt hiệu quả các phép toán di truyền. Ta xét bài toán tuyến tính bậc 2, minh họa những sự khác biệt giữa biểu diễn nhị phân và dấu chấm động.

Bài toán: Xét bài toán điều khiển tự động

Min $x^2{}_N + \sum_{k=0}^{N-1}(x^2{}_k + u^2{}_k)$ với $x_{k+1}=x_k+u_k$, k=0,1,...,N-1;

X_0 là trạng thái đầu được cho; $x_k \in R$ là biến trạng thái; $u \in R^N$ là vec to điều khiển.

Nghiệm giải tích của giá trị tối ưu là $J^* = K_0 x^2{}_0$ trong đó K_k là nghiệm của phương trình Riccati: $K_k=1+ K_{k+1}/(1+ K_{k+1})$ và $K_N=1$. Trong ví dụ, nhiễm sắc thể sẽ biểu diễn vecto trạng thái điều khiển u, miền cố định là (-200,200) cho mỗi u_i, chọn $x_0=100$ và N=45, nghĩa là nhiễm sắc thể $u=<u_0, ...u_{44}>$, có giá trị tối ưu $J^*=16180.4$.

3.3.2. Phiên bản cài đặt theo biểu diễn nhị phân

1. Biễu diễn nhiễn sắc thể: Mỗi phần tử của nhiễm sắc thể được biểu diễn bằng cùng một số bit, để mã hóa nhanh mỗi phần tử tương ứng với một từ nhớ, theo cách này có thể khai báo các phần tử thuộc kiểu integer.

Mỗi nhiễm sắc thể là một vecto gồm N từ nhó bằng với số phần tử của nhiễn sắc thể. Độ chính xác của cách tiếp cận này phụ thuộc vào số bit thực sự sử dụng và bằng $(UB-LB)/(2^n-1)$, trong đó UB và LB là các cận của miền xác định, còn n số bít trong mỗi phần tử của một nhiễm sắc thể. Xét quá trình Đột biến và lai tạo ngẫu nhiên:

Với biểu diễn nhị phân sử dụng những phép toán đột biến và lai tạo truyền thống, nhưng để giống với các phép toán của biểu diễn thực ta chỉ thực hiện lai giữa những phần tử, xác suất lai được giữ cố định là 0.25, trong khi xác suất đột biến được thay đổi để đạt được tốc độ cập nhật nhiễm sắc thể mong muốn:

Phiên bản	Xác suất cập nhật nhiễm sắc thể				
	0.6	0.7	0.8	0.9	0.95
Nhị phân, p_m	0.00047	0.00068	0.00098	0.0015	0.0021
Thực, p_m	0.014	0.02	0.03	0.045	0.061

2. Các phép toán khác:

Ngoài những phép toán đã được mô tả, có thể cài đặt thêm phép lai nhiều điểm và cũng cho phép lai trong số các bit của một phần tử. Phép lai nhiều điểm có xác suất áp dụng vào một phần tử được điều khiển bởi tham số hệ thống là 0.3.

3.4. Bài toán người du lịch (TSP-Traveling Salesman Problem)

Bài toán TSP: Một khách du lịch muốn đi thăm n thành phố T_1, T_2, ...T_n xuất phát từ một thành phố bất kỳ, mỗi thành phố đúng một lần, rồi trở về điểm khởi hành. Chi phí du lịch giữa từng cặp thành phố được cho trước bởi ma trận C. Hãy xây dựng hành trình cho khách du lịch sao cho tổng chi phí cho toàn hành trình là tối thiểu?

Như vậy, không gian tìm kiếm của TSP là tập các hoán vị của n thành phố, bất cứ hoán vị nào của n thành phố cũng là 1 lời giải chấp nhận được (là toàn bộ một hành trình qua các thành phố theo yêu cầu). Lời giải tối ưu là 1 hoán vị với chi phí tối thiểu của hành trình.

- Kích thước của không gian tìm kiếm là n!
- TSP là 1 bài toán tương đối cổ điển xuất hiện năm 1759 bởi nhà khoa học Euler. TSp được tập đoàn RAND giới thiệu vào năm 1948. TSP được chứng minh thuộc loại NP - khó. Nó xuất hiện trong nhiều ứng dụng và số thành phố là hoàn toàn có ý nghĩa.
- Có khá nhiều giải thuật toán đạt gần đến lời giải tối ưu của bài toán TSP. Giải thuật láng giềng gần nhất, thuật giải tham lam, đảo gần nhất, xa nhất, ..., các giải thuật do Like, Crap....
- TSP cũng có nhiều giải thuật di truyền tiếp cận tới, các các giải thuật này nhằm tạo ra các lời giải gần tối ưu bằng cách duy trì 1 quần thể các lời giải trải qua những biến đổi một ngôi, hai ngôi (đột biến & lai tạo) theo một kế hoạch chọn lọc nhiều về phía những cá thể phù hợp.
- Với giải thuật di truyền giải bài toán TSP thì biểu diễn nhị phân của các hành trình không thích hợp với TSP. (Kết quả cuối cùng ta cần có 1 hoán vị tốt nhất của các thành phố (i_1, i_2, .., i_n) trong đó (i_1, i_2, .., i_n) là hoán vị của {1, 2, ..., n}, mã nhị phân cho những thành phố này sẽ không thuận

lợi. Ngược lại, biểu diễn nhị phân cần đến các giải thuật sửa chữa đặc biệt, do thay đổi của một bit duy nhất cũng có thể đưa đến 1 hành trình bất hợp lệ).

- Ta xét 3 biểu diễn vecto khi giải TSP bằng giải thuật di truyền: Biểu diễn kề, thứ tự và đường dẫn

1.Biểu diễn kề

Biểu diễn kế biểu diễn một hành trình là danh sách n thành phố. Thành phố j được đặc trưng trong danh sách ở vị trí i nếu và chỉ nếu chuyến đi bắt đầu từ thành phố i đến thành phố j

Ví dụ: vector x=(2 4 8 3 9 7 1 5 6) sẽ biểu diễn hành trình sau đây:

1-> 2-> 4 -> 3 -> 8 -> 5 -> 9 -> 6 -> 7 -> 1 (*)

Vì:

1	2	3	4	5	6	7	8	9
2	4	8	3	9	7	1	5	6

x[1]=2 : 1 -> 2

x[2]=4: 2 -> 4

x[3]=8 : 3 -> 8

x[4]=3 : 4 -> 3

x[5]= 9 : 5 -> 9

x[6]=7 : 6 -> 7

x[7]=1 : 7 -> 1

x[8]=5 : 8 -> 5

x[9]=6 : 9 -> 6

Ghép lại ta có hành trình (*)

Mỗi thành phố chỉ xuất hiện 1 lần trong danh sách kề, nhưng 1 số danh sách kế có thể biểu diễn các hành trình bất hợp lệ (nghĩa là một phần hành trình có 1 chu trình (sảy ra quá sớm))

Ví dụ:Vector x=(2 4 8 1 9 3 5 7 6)

x[1]=2 : 1 -> 2

x[2]=4: 2 -> 4

x[3]=8 : 3 -> 8

x[4]=1 : 4 -> 1

x[5]= 9 : 5 -> 9

x[6]=3: 6 ->3

x[7]=5 : 7 -> 5

x[8]=7 : 8 -> 7

x[9]=6 : 9 -> 6

Hành trình: 1-> 2-> 4 -> 1

Biểu diễn kề không hỗ trợ toán tử lai cổ điển, cần một giải thuật sửa chữa

Ta xét 3 toán tử lai được định nghĩa cho biểu diễn kế:

+ Lai có cạnh thay đổi

+ Lai các đoạn hành trình con

+ Lai Heuristic

a.Lai cạnh thay đổi: tạo ra con mới bằng cách chọn (ngẫu nhiên) một cạnh cha – mẹ thứ nhất, rồi chọn một cạnh thích hợp từ cha – mẹ thứ hai, v.v… Toán tử mở rộng hành trình bằng cách chọn các cạnh từ các cha – mẹ luân phiên. Nếu cạnh mới từ một trong cha – mẹ tạo một chu trình vào hành trình hiện hành (mới một phần) thì toán tử chọn một cạnh thay thế (ngẫu nhiên) từ các cạnh còn lại không tạo các chu trình

Ví dụ: Từ 2 cha – mẹ:

P1 = (2 3 8 7 9 1 4 5 6) Chu trình sớm: 1 -> 2 -> 3-> 8 -> 5 -> 9 -> 6 -> 1

Và P2= (7 5 1 6 9 2 8 4 3) 1 -> 7 -> 8-> 4 -> 6 -> 2 -> 5 -> 9 -> 3 -> 1

Con thứ nhất có thể là: O_1 =(2 5 8 7 9 1 6 4 3)

Chu trình: 1 -> 2 -> 5-> 9 -> 3 -> 8 -> 4 -> 7 -> 6 -> 1

Ở đây tiến trình khởi đầu từ cạnh (1, 2) của cha – mẹ P1, cạnh ngẫu nhiên duy nhất được tạo trong tiến trình xen kẽ các cạnh là (7, 6) thay cho (7,8) sẽ tạo ra một chu trình được chấp nhận.

b. Lai các đoạn hành trình con: tạo ra con mới bằng cách chọn hành trình con (chiều dài ngẫu nhiên) từ một trong các cha – mẹ, rồi chọn hành trình con (chiều dài ngẫu nhiên) từ một trong các cha – mẹ khác, v.v… toán tử mở rộng hành trình bằng cách chọn các cạnh từ cha – mẹ luân phiên. Nếu một cạnh nào đó (từ một trong các cha – mẹ) tạo 1 chu trình vào hành trình hiện hành (một phần mới) thì toán tử chọn môt cạnh thay thế (ngẫu nhiên) từ các cạnh còn lại không tạo các chu trình.

c. Lai Heuristic: tạo ra con mới bằng cách chọn một hành trình ngẫu nhiên làm điểm khởi hành cho hành trình của con, rồi so sánh 2 cạnh (từ cả 2 cha - mẹ) rời thành phố này và chọn cạnh (ngắn hơn) tốt hơn.

Thành phố ở đầu bên kia của cạnh được chọn dùng làm điểm khởi hành trong việc chọn cạnh ngắn hơn trong hai cạnh rời thành phố này v.v…

Nếu ở một giai đoạn nào đó, cạnh mới đưa một chu trình vào hành trình thì hành trình sẽ được mở rộng bởi một cạnh ngẫu nhiên từ các cạnh còn lại không tạo chu trình.

Tác động của toán tử này là để dán những đường dẫn con ngắn của các hành trình cha – mẹ. Nhưng nó có thể để lại những giao điểm không muốn có của các cạnh, đó chính là lý do mà lai – Heuristic không thích hợp cho việc dò tìm chính xác các hành trình. Hai nhà khoa học Suh và Gucht đưa vào một toán tử Heuristic bổ sung thích hợp cho việc dò tìm chính xác

Toán tử này chọn ngẫu nhiên 2 cạnh (i, j) và (k, m) sau đó kiểm tra:

$d(i,j) + d(k,m)>d(i,m) + d(k,j)$

với d(a,b) là khoảng cách giữa 2 thành phố a & b. Nếu đây là trường hợp các cạnh (i,j) và (k, m) thì trong hành trình được thay thế bởi các cạnh (i, m) là (k, j).

Nhận xét:

- Lại các cạnh xen kẽ thường làm rối loạn các hành trình tốt do thao tác của chính nó qua các cạnh xen kẽ từ 2 cha – mẹ.

- Lai đoạn hành trình con thực hiện tốt hơn lai các cạnh xen kẽ do tỉ lệ rối loạn thấp hơn nhưng kết quả vẫn rất thấp.

- Lai Heuristic là toán tử tốt nhất ở đây vì 2 phép lai đầu tiên là mù quáng, nghĩa là chúng không để ý đến các chiều dài của các cạnh vì vậy nó thực hiện tốt hơn 2 cách kia. Nhưng kết quả của lai Heuristic vẫn chưa phải là xuất sắc.

2. Biểu diễn thứ tự: Biểu diễn hành trình là danh sách của n thành phố, phần tử thứ i của danh sách là một số trong khoảng từ 1 đến n-i+1.

Ý nghĩa của diều diễn thứ tự như sau:

+ Có 1 danh sách thứ tự của các thành phố C được dùng làm điểm tham chiếu cho các danh sách trong biểu diễn thứ tự:

Ví dụ: cho danh sách của các thành phố C = (1 2 3 4 5 6 7 8 9)

Một hành trình 1 – 2 - 4 – 3 – 8 – 5 – 9 – 6 – 7

Được biểu diễn là danh sách L của các tham chiếu:

L=(1 1 2 1 4 1 3 1 1) được hiểu như sau:

+ Số thứ nhất trong danh sách l là 1, vì vậy lấy thành phố thứ nhất trong danh sách C là thành phố thứ nhất của hành trình (thành phố số 1) sau đó lấy 1 ra khỏi C. Hành trình 1 phần là 1; C=(2 3 4 5 6 7 8 9)

+ Số kế tiếp trong danh sách là 1, vì thế lấy thành phố thứ nhất trong danh sách C hiện hành làm thành phố kế tiếp của hành trình (thành phố số 2), và lấy ra khỏi C. Hành trình 1 phần là 1 – 2; C=(3 4 5 6 7 8 9)

+ Số tiếp theo trong danh sách l là 2 vì thế lấy thành phố thứ 2 trong C là thành phố thứ 4 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4; C=(3 5 6 7 8 9)

+ Số kế tiếp trong danh sách là 1, vì thế thành phố 3 làm thành phố kế tiếp và lấy ra khỏi C

Hành trình thành phần là: 1 – 2 – 4 – 3; C=(5 6 7 8 9)

+ Số kế tiếp trong danh sách l là 4 vì thế lấy thành phố 8 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4 – 3 - 8; C=(5 6 7 9)

+ Số kế tiếp trong danh sách l là 1 vì thế lấy thành phố 5 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4 – 3 – 8 - 5; C=(6 7 9)

+ Số kế tiếp trong danh sách l là 3 vì thế lấy thành phố 9 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4 – 3 – 8 – 5 - 9; C=(6 7)

+ Số kế tiếp trong danh sách l là 1 vì thế lấy thành phố 7 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4 – 3 – 8 – 5 – 9 – 6; C=(7)

+ Số kế tiếp trong danh sách l là1 vì thế lấy thành phố 7 làm thành phố kế tiếp của hành trình và lấy ra khỏi C.

Hành trình thành phần là: 1 – 2 – 4 – 3 – 8 – 5 – 9 – 7

• Ưu điểm: của biểu diễn thứ tự là phép lai cổ điển hoạt động khá tốt. Hai hành trình bất kỳ trong biểu diễn thứ tự được cắt sau một vị trí nào đó và giao nhau, sẽ tạo ra 2 con, mỗi con là 1 hành trình hợp lệ.

Ví dụ: 2 cha – mẹ P_1=(1 1 2 1| 4 1 3 1 1) và P_2=(5 1 5 5| 5 3 3 2 1)

Tương ứng với hành trình 1 – 2 – 4 – 3 | 8 – 5 – 9 – 6 – 7

5 – 1 – 7 – 8 | 9 – 4 – 6 – 3 – 2

Với điểm lai tạo được đánh dấu bởi "|", tạo ra các con sau đây:

O_1=(1 1 2 1| 5 3 3 2 1); O_2 =(5 1 5 5 4 1 3 11)

Các con này ứng với các hành trình: 1 – 2 – 4 – 3 || - 9 – 7 – 8 – 6 – 5

Và 5 – 1 – 7 – 8 || - 6 – 2 – 9 – 3 – 4

Với kết quả O_1, O_2 ta thấy các hành trình một phần không thay đổi phía bên trái điểm lai, trong khi các hành trình một phần bên phải điểm lai lại bị rối loạn theo cách hoàn toàn ngẫu nhiên

(Các kết quả cho thấy biểu diễn này cùng phép lai cổ điển không thích hợp cho TSP)

3. Biểu diễn đường dẫn: là biểu diễn tự nhiên nhất của một hành trình.

Ví dụ: Hành trình 5 – 1 – 7 – 8 – 9 – 4 – 6 – 2 – 3, đơn giản được biểu diễn là: (5 1 7 8 9 4 6 2 3)

- Có 3 phép lai được định nghĩa cho biểu diễn đường dẫn:

1 – Lai – được ánh xạ từng phần (PXM)

2 – Lai – thứ tự (OX)

3 – Lai – chu trình (CX)

3.1.Lai PMX: do Golberg và Lingele đề xuất:

Tạo ra con mới bằng cách chọn một chuỗi con của hành trình từ một cha – mẹ đồng thời bảo toàn thứ tự và vị trí của tối đa có thể các thành phố từ cha – mẹ kia. Một chuỗi con của hành trình được chọn bằng cách chọn 2 điểm cắt ngẫu nhiên được dùng làm giới hạn cho các thao tác hoán vị.

Ví dụ: 2 cha – mẹ (với 2 điểm cắt được đánh dấu bởi “|”)

P1= (1 2 3 | 4 5 6 7 | 8 9) và

P2=(4 2 5 | 1 8 7 6 | 9 3)

Tạo ra con theo cách sau: trước hết các đoạn giữa các điểm cắt được hoán vị (“x” được hiểu là hiện tại chưa biết) Có 2 con:

O_1= (x x x|1 8 7 6 |x x)

O_2=(x x x|4 5 6 7 |x x)

Hoán vị này định nghĩa một loạt các ánh xạ 1 <->4; 8<->5; 7<->6; 6<->7

Ta thêm vào các thành phố (từ các cha mẹ gốc) mà không có xung đột:

O_1 = (x 2 3 | 18 7 6 | x 9) và

O_2 = (x x 2| 4 5 6 7 | 9 3)

Ta thấy x đầu tiên trong con O_1 (lẽ ra là 1 nhưng có xung đột) được thay bằng 4 (do ánh xạ 1<->4; x còn lại trong O_1(lẽ ra là 8 nhưng có xung đột) được thay bằng 5 (do ánh xạ 8<->5)

Tương tự trong O_2: 4 được thay bằng 1, 5 được thay bằng 8

Do đó có các con là O_1 = (4 2 3| 1 8 7 6 | 5 9) và

O_2 = (1 8 2|4 5 6 7 |9 3)

Kết luận: lai PMX khai thác các điểm tương đồng quan trọng trong giá trị và xếp bậc đồng thời khi được sử dụng với một kế hoạch sinh sản thích hợp.

3.2.Lai OX – Do Davis đề xuất: tạo ra các con bằng cách chọn một chuỗi con của hành trình từ một cha- mẹ và bảo tồn thứ tự tương đối của các thành phố của cha – mẹ kia.

Ví dụ: hai cha mẹ (với 2 điểm cắt được đánh dấu bởi ‘|’)

P1= (1 2 3 | 4 5 6 7 | 8 9) và P2=(4 2 5 | 1 8 7 6 | 9 3)

Tạo ra các con như sau:

+ Đầu tiên, các đoạn giữa các điểm cắt được sao chép vào các con:

O_1=(x x x|4 5 6 7 |x x)

O_2=(x x x|1 8 7 6 |x x)

Tiếp theo, bắt đầu từ điểm cắt thứ 2 của một cha – mẹ, các thành phố từ cha – mẹ kia được sao chép theo cùng thứ tự, bỏ đi các dấu hiệu đã có. Đến cuối chuỗi, ta tiếp tục từ vị trí đầu tiên của chuỗi.

9 – 3 – 4 – 5 – 2 – 1 – 8 – 7 – 6

Sau đó bỏ đi các thành phố 4, 5, 6, 7 đã có trong con thứ nhất O_1, ta có:

9 – 3 – 2 – 1 – 8

Thứ tự này ta được đặt vào con thứ nhất (bắt đầu từ điểm cắt thứ 2)

O_1=(2 1 8|4 5 6 7 | 9 3)

Tương tự ta có con O_2 là: O_2= (3 4 5|1 8 7 6 | 9 2) vì

8 – 9 – 1 – 2 – 3 – 4 – 5 – 6 – 7 bỏ đi 1, 8, 7, 6 còn lại là 9 – 2 – 3 – 4- 5 đưa vào O_2 bắt đầu từ điểm cắt thứ 2 ta có:

O_2= (3 4 5 | 1 8 7 6| 9 2)

Lai tạo kiểu OX khai thác thuộc tính về biểu diễn đường dẫn, mà thứ tự của các thành phố (chứ không phải vị trí của chúng) rất quan trọng, nghĩa là, lai hành trình: 9 – 3 – 4 – 5 – 2 – 1 – 8 – 7 – 6 và 4 – 5 – 2 – 1 – 8 – 7 – 6 – 9 – 3 thực ra là giống nhau.

3.3. Lai CX: Do Olive đề xuất. xây dựng con theo cách mỗi thành phố (và vị trí của nó) xuất phát từ một trong các cha – me. Ta xét cơ chế lai chu trình theo ví dụ sau:

Hai cha – mẹ: P_1= (1 2 3 4 5 6 7 8 9); P_2=(4 1 2 8 7 6 9 3 5)

Sẽ tạo ra con thứ nhất bằng cách lấy thành phố thứ nhất thừ cha – mẹ thứ nhất: O_1=$(1\ x\ \ x\ x\ x\ x\ x\ x)$

Do mỗi thành phố trong con có thể bị lấy đi từ một trong các cha mẹ của nó (từ cùng vị trí) nên ta sẽ lựa chọn thành phố kế tiếp được xét phải là thành phố 4, do thành phố từ cha-mẹ p_2 ngay bên dưới thành phố 1 được chọn. Trong P_1 thành phố này ở tại ví trí thứ 4, như vậy O_1=(1 x x 4 x x x x x)

Như vậy tiếp theo thành phố 8 được chọn, do thành phố từ cha – mẹ P_2 ngay bên dưới thành phố 4 được chọn, như vậy O_1=$(1\ x\ x\ 4\ x\ x\ x\ 8\ x)$

Tiếp theo thành phố 3 đươc chọn O_1=$(1\ x\ 3\ 4\ x\ x\ x\ 8\ x)$

Tiếp theo thành phố 2 đươc chọn O_1=$(1\ x\ 3\ 4\ x\ x\ x\ 8\ x)$

Sau khi chọn xong thành phố 2 tiếp theo đòi hỏi chọn thành phố 1 đã có trong danh sách, như vậy ta đã hoàn thành một chu trình:

$O_1=(1\ 2\ 3\ 4\ x\ x\ x\ 8\ x)$; Các thành phố khác được hoàn thành từ cha – mẹ kia $O_1=(1\ 2\ 3\ 4\ 7\ 6\ 9\ 8\ 5)$

Tương tự ta có: $O_2=(4\ x\ x\ x\ x\ x\ x\ x\ x)$ $O_2=(4, 1, 2, 8, 5, 6, 7, 3, 9)$

Kết luận: Lai trên vị trí (Lai chu trình) giống với lai thứ tự gốc nhiều hơn, khác biệt duy nhất là trong lai trên vị trí, thay vì chọn 1 chuỗi con các thành phố để sao chép thì nhiều thành phố lại được chọn (ngẫu nhiên) cho mục đích này.

Các phương pháp gần đây (các chiến lược tiến hóa) để giải bài toán TSP, xét thử nghiệm với 4 toán tử đột biến:

1. Đảo

2.Chèn- chọn 1 thành phố và chèn vào 1 vị trí ngẫu nhiên

3. Đổi chỗ - chọn 1 hành trình con rồi chèn nó vào một vị trí ngẫu nhiên

4. Trao đổi qua lại – hoán vị 2 thành phố

1. Toán tử đảo: Phép đảo đơn giản chọn 2 điểm theo chiều dài của NSthể, được cắt tại những điểm này và chuỗi con giữa các điểm này bị đảo.

Ví dụ: Nhiễm sắc thể: $\langle 1\ 2\ 3|4\ 5\ 6\ 7|8\ 9\rangle$

Với 2 điểm cắt được đánh dấu bởi ‘|’được đổi thành: $\langle 1\ 2|6\ 5\ 4\ 3|7\ 8\ 9\rangle$

Đảo đơn giản nhưng bảo đảm rằng đứa con có được một hành trình hợp lệ.

Một toán tử lai Heuristic cũng được sử dụng với nhiều cha – mẹ góp sức trong việc tạo ra con. Sau khi chọn xong thành phố đầu tiên trong hành trình của các con (ngẫu nhiên) tất cả các lân cận bên phải và trái của thành phố đó (từ tất cả cha – mẹ) đều được khảo sát. Thành phố có khoảng cách ngắn nhất sẽ được chọn. Tiến trình tiếp tục cho đến khi hoàn tất hành trình.

Hầu hết cá toán tử được xét đến từ trước tới nay đều quan tâm đến các thành phố, nghĩa là các vị trí và thứ tự của chúng. Ngược lại với các cạnh (cung nối giữa các thành phố) vị trí cụ thể của một thành phố trong hành trình không quan trọng mà là cách nối giữa thành phố này với các thành phố khác .

Grefenstette đã phát triển một lớp các toán tử heuristic chú trọng đến các cạnh, các toán tử này làm việc như sau:

1. Chọn ngẫu nhiên thành phố của các con làm thành phố C hiện hành.
2. Chọn 4 cạnh (chọn 2 cạnh từ từng cha - mẹ) gắn liền với thành phố hiện hành C.
3. Xác định phân bố xác suất trên các cạnh được chọn, dựa trên chi phí của chúng. Xác suất của cạnh có liên quan đến thành phố được thăm trước đây là 0.
4. Chọn 1 cạnh, nếu ít nhất một cạnh có xác suất khác 0, việc chọn được dựa vào phân bố ở trên; Ngược lại, chọn ngẫu nhiên (từ những thành phố chưa thăm).
5. Thành phố ở "đầu kia" của cạnh được chọn trở thành thành phố hiện hành C
6. Nếu hành trình hoàn tất, dừng lại; Ngược lại, chưa hoàn tất ta quay trở lại bước 2

Với thủ tục này ta chuyển được khoảng 60% các cạnh từ các cha - mẹ còn 40% các cạnh được chọn ngẫu nhiên.

Whitley, Starweather và Fuquay đã phát triển một toán tử lai tạo mới: Lai tạo tái kết hợp cạnh (ER), chuyển hơn 95% các cạnh từ cha - mẹ cho đứa con duy nhất. Toán tử ER khai thác thông tin trên các cạnh trong một hành trình.

Xét ví dụ sau:Hành trình: (3 1 2 8 7 4 6 9 5)

Các cạnh là (3,1); (1,2); (2,8); (8,7); (7,4); (4,6); (6,9); (9,5); (5,3)

Các cạnh (không phải là thành phố) mang các giá trị (các khoảng cách) trong TSP.

Hàm mục tiêu cần cực tiểu hoá là tổng các cạnh xây dựng một hành trình hợp lệ. Vị trí của một thành phố trong hành trình không quan trọng, các hành trình là khép kín, hướng của một cạnh cũng không quan trọng: cả 2 cạnh (3,1) và (1,3) đều chỉ cho thấy rằng các thành phố 1 và 3 được nối trực tiếp.

Tư tưởng chính của lai ER là một con duy nhất được xây dựng từ các cạnh hiện diện trong cả 2 cha - mẹ. Điều này được thực hiện với sự trợ giúp của danh sách cạnh được tạo ra từ các hành trình của cả 2 cha mẹ. Danh sách cạnh cung cấp cho mỗi thành phố C tất cả các thành phố khác nối với thành phố C trong ít nhất là một cha - mẹ. Như vậy có ít nhất là 2 và nhiều nhất là 4 thành phố trong danh sách cho mỗi thành phố C.

Ví dụ: Với 2 cha - mẹ: P_1= (1 2 3 4 5 6 7 8 9) và P_2= (4 1 2 8 7 6 9 3 5)

Danh sách cạnh là :

Thành phố	Có cạnh nối đến các thành phố
1	9, 2, 4
2	1, 3, 8
3	2, 4, 9, 5
4	3, 5, 1
5	4, 6, 3
6	5,7, 9
7	6, 8
8	7, 9, 2
9	8, 1, 6, 3

Việc xây dựng con được bắt đầu bằng việc chọn 1 thành phố khởi tạo từ 1 cha - mẹ, ví dụ thành phố 1 hoặc 4. Thành phố có số cạnh nhỏ nhất trong danh sách cạnh sẽ được chọn, nếu những số này bằng nhau thì chọn ngẫu nhiên. Việc chọn như thế sẽ làm tăng cơ hội hoàn thành hành trình với tất cả các cạnh được chọn từ các cha mẹ. Với việc chọn ngẫu nhiên, cơ hội để xảy ra thất bại (ở bên trái với 1 thành thành phố không có cạnh đi tiếp) sẽ cao hơn nhiều. Giả sử đã chọn một thành phố 1, thành phố này được nối trực tiếp với 3 thành phố khác là 9, 2 và 4, thành phố kế tiếp sẽ được chọn từ 3 thành phố này. Các thành phố 4 và 2 có 3 cạnh còn thành phố 9 có 4 cạnh => có 1 lựa chọn ngẫu nhiên giữa các thành phố 4 và 2, giả sẻ thành phố được chọn là 4, các ứng cử viên của thành phố kế tiếp trong hành trình xây dựng là 3 và 5, => thành phố 5 được chọn vì nó chỉ có 3 cạnh còn thành phố 3 có 4 cạnh.

Như vậy có dạng: O=(1 4 5 x x x x x x)

Tiếp tục thủ tục này ta được con: O=(1 4 5 6 7 8 2 3 9)

Con này được tạo hoàn toàn từ các cạnh lấy từ 2 cha mẹ (tỉ lệ thất bại cho việc chọn cạnh là rất thấp từ 1% đến 1,5%)

Nhận xét: Các toán tử tái kết hợp cạnh cho thấy biểu diễn đường dẫn có thể quá xấu để biểu diễn các thuộc tính quan trọng của một hành trình do đó người ta liên tục đi tìm các biểu diễn khác không phải là vector với những toán tử di truyền tương ứng phù hợp.

Phương pháp do Homaifar và Guau đề xuất:

Ma trận $M=(m_{ij})=1 \Leftrightarrow$ có 1 cạnh từ thành phố I đến thành phố j, và dung các toán tử lai tạo khác và đảo Heuristic.

Hai toán tử lai ma trận (MX) được định nghĩa như sau: những toán tử này trao đổi tất cả các mục nhập của 2 ma trận cha - mẹ hoặc sau 1 điểm lai tạo đơn (lai tạo một điểm) hay giữa 2 điểm lai tạo (lai tạo hai điểm).

Một giải thuật sửa chữa được bổ sung để thực hiện:

(1) Loại các chỗ trùng nhau (nghĩa là bảo đảm rằng mỗi hàng và mỗi cột chỉ có chính xác một số 1 và

(2) Cắt và nối các chu trình (nếu có) nhằm tạo ra một hành trình hợp lệ. Lai 2 điểm được xét trong ví dụ sau: Hai ma trận cha - mẹ cho trong hình 2a + 2b biểu diễn 2 hành trình hợp lệ. P_1: (1 2 4 3 8 6 5 7 9); P_2:(1 4 3 6 5 7 2 8 9)

	1	2	3	4	5	6	7	8	9
1			**0**	**1**	**0**	**0**			
2			**0**	**0**	**0**	**0**		1	
3			**0**	**0**	**0**	**1**			
4			**1**	**0**	**0**	**0**			
5			**0**	**0**	**0**	**0**	1		
6			**0**	**0**	**1**	**0**			
7		1	**0**	**0**	**0**	**0**			
8			**0**	**0**	**0**	**0**			1
9	1		**0**	**0**	**0**	**0**			

Hình2b

	1	2	3	4	5	6	7	8	9
1	0	1	**0**	**0**	**0**	**0**	0	0	0
2	0	0	**0**	**1**	**0**	**0**	0	0	0
3	0	0	**0**	**0**	**0**	**0**	0	1	0
4	0	0	**1**	**0**	**0**	**0**	0	0	0
5	0	0	**0**	**0**	**0**	**0**	1	0	0
6	0	0	**0**	**0**	**1**	**0**	0	0	0
7	0	0	**0**	**0**	**0**	**0**	0	0	1
8	0	0	**0**	**0**	**0**	**1**	0	0	0
9	1	0	**0**	**0**	**0**	**0**	0	0	0

Hình 2a

	1	2	3	4	5	6	7	8	9
1	0	1	**0**	**1**	**0**	**0**	0	0	0
2	0	0	**0**	**0**	**0**	**0**	0	0	0
3	0	0	**0**	**0**	**0**	**1**	0	1	0
4	0	0	**1**	**0**	**0**	**0**	0	0	0
5	0	0	**0**	**0**	**0**	**0**	1	0	0
6	0	0	**0**	**0**	**1**	**0**	0	0	0
7	0	0	**0**	**0**	**0**	**0**	0	0	1
8	0	0	**0**	**0**	**0**	**0**	0	0	0
9	1	0	**0**	**0**	**0**	**0**	0	0	0

	1	2	3	4	5	6	7	8	9
1	0	0	**0**	**0**	**0**	**0**	0	0	0
2	0	0	**0**	**1**	**0**	**0**	0	1	0
3	0	0	**0**	**0**	**0**	**0**	0	0	0
4	0	0	**1**	**0**	**0**	**0**	0	0	0
5	0	0	**0**	**0**	**0**	**0**	1	0	0
6	0	0	**0**	**0**	**1**	**0**	0	0	0
7	0	1	**0**	**0**	**0**	**0**	0	0	0
8	0	0	**0**	**0**	**0**	**1**	0	0	1
9	1	0	**0**	**0**	**0**	**0**	0	0	0

Hình 2a1

Hình 2b1

Hai điểm lai được chọn: + Điểm giữa cột 2 và cột 3 (điểm đầu trên)

+ Điểm giữa cột 6 và cột 7 (điểm thứ 2)

Các điểm lai cắt ma trận theo chiều dọc với mỗi ma trận, 2 cột đầu tiên lập thành phần đầu, các cột 3,4,5,6 thuộc phần giữa, 3 cột cuối 7, 8, 9 thuộc phần thứ 3. Các mục nhập của 2 ma trận được hoán đổi giữa các điểm lai. Kết quả trung gian của 2 con lai ở hình $2a_1$ và $2b_1$.

Cả 2 con ($2a_1$) và ($2b_1$) đều bất hợp lệ, nhưng tổng số các số 1 trong mỗi ma trận đều đúng bằng 9. Bước đầu tiên của giải thuật sửa chữa là di chuyển một vài số 1 trong ma trận để mỗi hàng và mỗi cột chỉ có đúng một con 1.

Ví dụ: Trong con $2a_1$, các số 1 trùng lặp xuất hiện ở hàng 1 và 3. Giải thuật có thể chuyển ô m_{14}=1 vào m_{84}; ô m_{38}=1 vào m_{28}

Trong con $2b_1$, các số 1 trùng lặp xuất hiện ở hàng 2 và 8. Giải thuật có thể chuyển ô m_{14}=1 vào m_{34}; Ô m_{86}=1 vào ô m_{16} Sau khi hoàn tất bước đầu tiên của giải thuật sửa chữa, con thứ nhất biểu diễn một hành trình hợp lệ: (1 2 8 4 3 6 5 7 9) và con thứ 2 biểu diễn hành trình con: (1 6 5 7 2 8 9) và (3 4)

Bước thứ 2 của giải thuật sửa chữa chỉ áp dụng cho con thứ 2. Ở đây, giải thuật cắt & nối các hành trình con để cho ra 1 hành trình hợp lệ, việc cắt và nối này xét đến các cạnh đang có trong cha - mẹ gốc.

Ví dụ: Cạnh (2, 4) được chọn để nối 2 hành trình con này vì cạnh này hiện có trong một cha - mẹ. Khi đó ta có hành trình trọn vẹn hợp lệ của con thứ 2 là: (1 6 5 7 2 4 3 8 9). Một chương trình tiến hóa tốt cho bài toán TSP cần kết hợp với các toán tử cải thiện cục bộ (nhóm đột biến), dựa trên các giải thuật dùng cho tối ưu hóa cục bộ, cùng các toán tử nhị phân được thiết kế cẩn thận, sẽ kết hợp thông tin Heuristic về bài toán.