Giới thiệu về thuật toán di truyền

Jenna Carr

Ngày 16 tháng 5 năm 2014

trừu tư ợng

Thuật toán di truyền là một loại thuật toán tối ư u hóa, nghĩa là chúng đư ợc sử dụng để
tìm cực đại hoặc cực tiểu của hàm số. Trong bài báo này chúng tôi giới thiệu, minh họa,
và thảo luận về các thuật toán di truyền cho ngư ời dùng mới bắt đầu. Chúng tôi cho thấy những thành phần nào tạo nên
thuật toán di truyền và cách viết chúng. Sử dụng MATLAB, chúng tôi lập trình một số
ví dụ, bao gồm một thuật toán di truyền để giải bài toán Ngư ời bán hàng du lịch cổ điển
Vấn đề. Chúng tôi cũng thảo luận về lịch sử của thuật toán di truyền, các ứng dụng hiện tại và
những phát triển trong tư ơ ng lai.

Thuật toán di truyền là một loại thuật toán tối ư u hóa, nghĩa là chúng đư ợc sử dụng để tìm (các) giải pháp tối ư u cho một vấn đề tính toán nhất định nhằm tối đa hóa hoặc tối thiểu hóa chức năng cụ thể. Các thuật toán di truyền đại diện cho một nhánh của lĩnh vực nghiên cứu đư ợc gọi là tính toán tiến hóa [4], trong đó chúng bắt chư ớc các quá trình sinh sản sinh học và chọn lọc tự nhiên để giải quyết các giải pháp 'phù hợp nhất' [1]. Giống như trong quá trình tiến hóa, nhiều Các quy trình của thuật toán di truyền là ngẫu nhiên, tuy nhiên kỹ thuật tối ư u hóa này cho phép ngư ời ta thiết lập mức độ ngẫu nhiên và mức độ kiểm soát [1]. Những thuật toán này còn hơ n thế nữa mạnh mẽ và hiệu quả hơ n các thuật toán tìm kiếm ngẫu nhiên và tìm kiếm toàn diện [4], tuy nhiên yêu cầu không có thêm thông tin về vấn đề nhất định. Tính năng này cho phép họ tìm giải pháp đối với các vấn đề mà các phư ơ ng pháp tối ư u hóa khác không thể giải quyết đư ợc do thiếu tính liên tục, đạo hàm, tính tuyến tính hoặc các đặc tính khác.

Phần 1 giải thích những gì tạo nên một thuật toán di truyền và cách chúng hoạt động. Phần 2 đi qua ba ví dụ đơn giản. Phần 3 trình bày lịch sử của thuật toán di truyền đã phát triển. Phần 4 trình bày hai bài toán tối ưu hóa kinh điển gần như không thể thực hiện được để giải quyết trước khi có sự ra đời của các thuật toán di truyền. Phần 5 thảo luận về cách các thuật toán này được sử dụng ngày nay.

1 Thành phần, cấu trúc và thuật ngữ

Vì các thuật toán di truyền được thiết kế để mô phỏng một quá trình sinh học, nên phần lớn các vấn đề liên quan thuật ngữ được mượn từ sinh học. Tuy nhiên, các thực thể mà thuật ngữ này đề cập đến trong các thuật toán di truyền đơn giản hơn nhiều so với các thuật toán sinh học của chúng [8]. Cơ bản Các thành phần chung của hầu hết các thuật toán di truyền là:

- chức năng tập thể dục để tối ưu hóa
- quần thể nhiễm sắc thể
- lựa chọn nhiễm sắc thể nào sẽ sinh sản
- lai ghép để tạo ra thế hệ nhiễm sắc thể tiếp theo
- đột biến ngẫu nhiên của nhiễm sắc thể ở thế hệ mới

Hàm thích nghi là hàm mà thuật toán đang cố gắng tối ưu hóa [8]. từ "sự phù hợp" được lấy từ thuyết tiến hóa. Nó được sử dụng ở đây vì các bài kiểm tra chức năng thể lực và định lượng mức độ 'phù hợp' của từng giải pháp tiềm năng. Chức năng tập thể dục là một trong những chức năng quan trọng nhất phần quan trọng của thuật toán, vì vậy nó sẽ được thảo luận chi tiết hơn ở cuối phần này. Các thuật ngữ nhiễm sắc thể đề cập đến một giá trị số hoặc các giá trị đại diện cho một giải pháp ứng cử viên cho vấn đề mà thuật toán di truyền đang cố gắng giải quyết [8]. Mỗi giải pháp ứng cử viên là được mã hóa dưới dạng một mảng các giá trị tham số, một quá trình cũng được tìm thấy trong các phương pháp tối ưu hóa khác thuật toán [2]. Nếu một bài toán có kích thước Npar thì thông thường mỗi nhiễm sắc thể sẽ được mã hóa

dư ới dạng mảng phần tử Npar

nhiễm sắc thể = [p1, p2, ..., pNpar]

trong đó mỗi pi là một giá trị cụ thể của i tham số [2]. Điều đó tùy thuộc vào ngư ời tạo ra thuật toán di truyền để nghĩ ra cách dịch không gian mẫu của các lời giải ứng viên thành nhiễm sắc thể. Một cách tiếp cận là chuyển đổi từng giá trị tham số thành một chuỗi bit (chuỗi của 1 và 0), sau đó nối các tham số từ đầu đến cuối giống như các gen trong chuỗi DNA tạo ra nhiễm sắc thể [8]. Trong lịch sử, nhiễm sắc thể thư ờng đư ợc mã hóa theo cách này, và nó vẫn là một phư ơ ng pháp phù hợp cho các không gian giải pháp rời rạc. Máy tính hiện đại cho phép nhiễm sắc thể bao gồm các hoán vị, số thực và nhiều đối tư ợng khác; như ng bây giờ chúng ta sẽ tập trung vào nhiễm sắc thể nhị phân.

Một thuật toán di truyền bắt đầu bằng một tập hợp các nhiễm sắc thể đư ợc chọn ngẫu nhiên, đóng vai trò là thế hệ đầu tiên (quần thể ban đầu). Khi đó mỗi nhiễm sắc thể trong quần thể đư ợc đánh giá bởi hàm thích hợp để kiểm tra xem nó giải quyết vấn đề hiện tại tốt như thế nào.

Bây giờ toán tử chọn lọc chọn một số nhiễm sắc thể để sinh sản dựa trên phân bố xác suất do ngư ời dùng xác định. Nhiễm sắc thể càng phù hợp thì càng có nhiều khả năng sẽ đư ợc chọn. Ví dụ: nếu f là hàm thích nghi không âm thì xác suất nhiễm sắc thể C53 đư ợc chọn để sinh sản có thể là

$$P(C53) = \frac{f(C53)}{\sum_{i=1}^{Npop} f(Ci)} .$$

Lư u ý rằng toán tử chọn lọc chọn nhiễm sắc thể có thay thế, do đó, cùng một nhiễm sắc thể mosome có thể đư ợc chọn nhiều lần. Toán tử chéo giống với toán tử sinh học trao đổi chéo và tái tổ hợp các nhiễm sắc thể trong quá trình phân bào. Toán tử này hoán đổi một chuỗi hai nhiễm sắc thể đư ợc chọn để tạo ra hai con. Ví dụ, nếu nhiễm sắc thể bố mẹ

[11010111001000] và [01011101010010]

được vượt qua sau bit thứ tư , sau đó

[01010111001000] và [11011101010010]

sẽ là con cháu của họ. Toán tử đột biến đảo ngẫu nhiên các bit riêng lẻ trong
nhiễm sắc thể (biến 0 thành 1 và ngược lại). Thông thường đột biến xảy ra với rất
xác suất thấp, chẳng hạn như 0,001. Một số thuật toán triển khai toán tử đột biến trước
các toán tử lựa chọn và chéo; đây là vấn đề ưu tiên. Thoạt nhìn,
toán tử đột biến có vẻ không cần thiết. Trên thực tế, nó đóng một vai trò quan trọng, ngay cả khi nó
là thứ yếu so với chọn lọc và lai ghép [1]. Lựa chọn và lai ghép duy trì
thông tin di truyền của nhiễm sắc thể phù hợp hơn, nhưng những nhiễm sắc thể này chỉ mang tính tương đối tốt hơn
đến thế hệ hiện tại. Điều này có thể khiến thuật toán hội tụ quá nhanh và mất
"vật liệu di truyền có khả năng hữu ích (1 hoặc 0 tại các vị trí cụ thể)" [1]. Nói cách khác,
thuật toán có thể bị kẹt ở mức tối ưu cục bộ trước khi tìm được mức tối ưu toàn cục [3].
Toán tử đột biến giúp bảo vệ chống lại vấn đề này bằng cách duy trì tính đa dạng trong
nhưng nó cũng có thể làm cho thuật toán hội tụ chậm hơn.

Thông thường quá trình chọn lọc, lai ghép và đột biến tiếp tục cho đến khi số lượng
thế hệ con cháu giống với quần thể ban đầu nên thế hệ thứ hai được tạo thành
hoàn toàn là con cái mới và thế hệ đầu tiên được thay thế hoàn toàn. Chúng ta sẽ thấy điều này
phương pháp trong ví dụ 2.1 và 2.2. Tuy nhiên, một số thuật toán cho phép các thành viên có mức độ phù hợp cao của
thế hệ đầu tiên sống sót sang thế hệ thứ hai. Chúng ta sẽ thấy phương pháp này trong Ví dụ 2.3
và Phần 4.

Bây giờ thế hệ thứ hai được kiểm tra bằng chức năng thích nghi và chu kỳ lặp lại. Nó là
một phương pháp phổ biến để ghi lại nhiễm sắc thể có độ thích hợp cao nhất (cùng với độ thích hợp của nó
value) từ mỗi thế hệ hoặc nhiễm sắc thể "tốt nhất cho đến nay" [5].

Các thuật toán di truyền được lặp lại cho đến khi đạt được giá trị thích hợp của nhiễm sắc thể "tốt nhất cho đến nay"
ổn định và không thay đổi qua nhiều thế hệ. Điều này có nghĩa là thuật toán đã hội tụ

đến (các) giải pháp. Toàn bộ quá trình lặp lại đư ợc gọi là chạy. Vào cuối mỗi lần chạy có thư ờng có ít nhất một nhiễm sắc thể là giải pháp rất phù hợp cho vấn đề ban đầu. Tùy thuộc vào cách viết thuật toán, đây có thể là thuật toán phù hợp nhất trong số tất cả các thuật toán "tốt nhất cho đến nay" nhiễm sắc thể, hoặc phù hợp nhất của thế hệ cuối cùng.

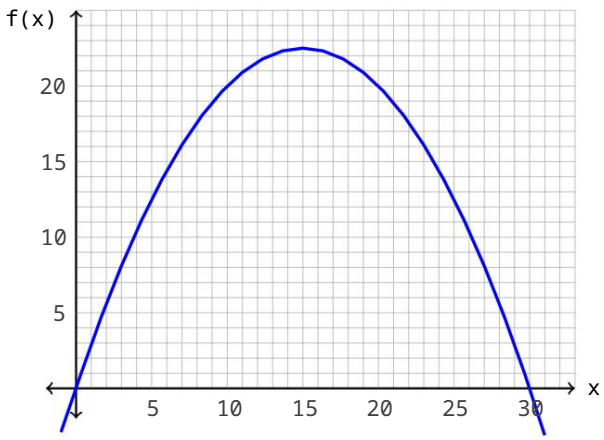
"Hiệu suất" của thuật toán di truyền phụ thuộc rất nhiều vào phư ơ ng pháp đư ợc sử dụng để mã hóa giải pháp ứng cử viên vào nhiễm sắc thể và "tiêu chí cụ thể để thành công" hoặc những gì chức năng tập thể dục thực sự đang đo lư ờng [7]. Các chi tiết quan trọng khác là xác suất trao đổi chéo, xác suất đột biến, quy mô quần thể và số lư ợng lần lặp lại. Các giá trị này có thể đư ợc điều chỉnh sau khi đánh giá hiệu suất của thuật toán trên vài lần chạy thử.

Các thuật toán di truyền đư ợc sử dụng trong nhiều ứng dụng. Một số ví dụ nổi bật là lập trình tự động và học máy. Họ cũng rất phù hợp với việc làm ngư ời mẫu hiện tư ợng trong kinh tế, sinh thái, hệ thống miễn dịch của con ngư ời, di truyền dân số và các hệ thống xã hội.

1.1 Lư u ý về chức năng thể dục

Tiếp tục sự tư ơ ng tự của chọn lọc tự nhiên trong tiến hóa sinh học, hàm thích ứng giống như môi trư ờng sống mà sinh vật (các giải pháp ứng cử viên) thích nghi. Đó là bư ớc duy nhất trong thuật toán xác định các nhiễm sắc thể sẽ thay đổi như thế nào theo thời gian và có thể có nghĩa là sự khác biệt giữa việc tìm ra giải pháp tối ư u và không tìm thấy giải pháp nào cả. Kinnear, biên tập viên của tạp chí Những tiến bộ trong lập trình di truyền, giải thích rằng "chức năng thể dục là chức năng duy nhất cơ hội là bạn phải truyền đạt ý định của mình với quá trình mạnh mẽ mà di truyền lập trình đại diện. Hãy đảm bảo rằng nó truyền đạt chính xác những gì bạn mong muốn" [4]. Nói một cách đơ n giản, "bạn không thể quá quan tâm đến việc chế tạo" nó [4]. Kinnear nhấn mạnh rằng sự tiến hóa của dân số sẽ "khai thác một cách tàn nhẫn" mọi "điều kiện biên giới" và tinh vi khiếm khuyết trong chức năng thích nghi [4] và cách duy nhất để phát hiện điều này là chỉ chạy thuật toán và kiểm tra các nhiễm sắc thể thu đư ợc.

Hình 1: Đồ thị hàm số $f(x) = \frac{x^2}{10} + 3x$



Chức năng tập thể dục phải nhạy hơn việc chỉ phát hiện đâu là nhiễm sắc thể 'tốt'. một số so với nhiễm sắc thể 'xấu': nó cần chấm điểm chính xác các nhiễm sắc thể dựa trên phạm vi các giá trị thích hợp, sao cho có thể phân biệt được một giải pháp hoàn chỉnh với một giải pháp giải pháp hoàn thiện hơn [4]. Kinnear gọi việc trao giải thưởng này là “tín dụng một phần” [4]. Nó quan trọng để xem xét giải pháp từng phần nào nên được ưu tiên hơn các giải pháp từng phần khác vì điều đó sẽ quyết định hướng di chuyển của toàn bộ dân số [4].

2 ví dụ sơ bộ

Phần này sẽ xem xét một số ví dụ đơn giản về hoạt động của thuật toán di truyền. Họ được trình bày theo thứ tự độ phức tạp tăng dần và do đó giảm tính tổng quát.

2.1 Ví dụ: Tối đa hóa hàm một biến

Ví dụ này phỏng theo phương pháp của một ví dụ được trình bày trong cuốn sách của Goldberg [1].

Xét bài toán cực đại hóa hàm

$$f(x) = \frac{x^2}{10} + 3x$$

trong đó x được phép thay đổi trong khoảng từ 0 đến 31. Hàm này được hiển thị trên Hình 1. Để giải

Bằng cách sử dụng thuật toán di truyền, chúng ta phải mã hóa các giá trị có thể có của x dưới dạng nhiễm sắc thể. Vì

ví dụ này, chúng ta sẽ mã hóa x dưới dạng số nguyên nhị phân có độ dài 5. Do đó, nhiễm sắc thể của

Thuật toán di truyền của chúng tôi sẽ là các chuỗi 0 và 1 với độ dài 5 bit và có

nằm trong khoảng từ 0 (00000) đến 31 (11111).

Để bắt đầu thuật toán, chúng tôi chọn ngẫu nhiên quần thể ban đầu gồm 10 nhiễm sắc thể.

Chúng ta có thể đạt được điều này bằng cách tung một đồng xu công bằng 5 lần cho mỗi nhiễm sắc thể, để mặt ngửa biểu thị

1 và đuôi biểu thị 0. Quần thể nhiễm sắc thể ban đầu thu được được thể hiện trong Bảng

1. Tiếp theo, chúng tôi lấy giá trị x mà mỗi nhiễm sắc thể đại diện và kiểm tra mức độ phù hợp của nó với

chức năng tập thể dục. Các giá trị thích hợp thu được được ghi lại trong cột thứ ba của Bảng 1.

Bảng 1: Dân số ban đầu

Nhiễm sắc thể ban đầu		x	Sự thích hợp	Lựa chọn
Số Dân số	Giá trị	Giá trị f(x)	Xác suất	
1	0 1 0 1 1	11	20,9	0,1416
	1 1 0 1 0	26	10,4	0,0705
2	3	2	5,6	0,0379
	0 1 1 1	14	22,4	0,1518
4	5 0 0 1 1 0	12	21,6	0,1463
6	1 1 1 1 0	30		0
7	1 0 1 1 0	22	0	0,1192
8	0 1 0 0 1		17,6	0,1280
9	0 0 0 1 1			0,0549
10	1 0 0 0 1	9 3 17	18,9 8,1 22,1	0,1497
		Tổng	147,6	
		Trung bình	14,76	
		Tối đa	22,4	

Chúng tôi chọn các nhiễm sắc thể sẽ sinh sản dựa trên giá trị thể lực của chúng bằng cách sử dụng

xác suất sau:

$$P(\text{nhiễm sắc thể tôi tái tạo}) = \frac{f(x_i)}{\sum_{k=1}^{10} f(x_k)}$$

Goldberg ví quá trình này giống như việc quay một bánh xe roulette có trọng lượng [1]. Vì dân số của chúng ta có 10 nhiễm sắc thể và mỗi lần 'giao phối' tạo ra 2 con, chúng ta cần 5 lần giao phối để tạo ra một thế hệ mới gồm 10 nhiễm sắc thể. Các nhiễm sắc thể đã chọn được hiển thị trong Bảng 2. Để tạo ra con cái của chúng, một điểm giao nhau được chọn ngẫu nhiên, được hiển thị trong bảng như một đường thẳng đứng. Lưu ý rằng có thể sự giao thoa không xảy ra, trong trường hợp đó con cái là bản sao chính xác của cha mẹ chúng.

Bảng 2: Sinh sản & Thế hệ thứ hai

Giao phối nhiễm sắc thể Con số	Cặp	Mới Giá trị giá trị dân số f(x)	x	Sự thích hợp
	0 1|1 0 0	0 1 0 1	10	20
5 2	1 1|0 1 0	0 1 1 1 0 0	28	5,6
4	0 1 1 1|0	0 1 1 1	15	22,5
8	0 1 0 0|1	1 0 1 0 0		17,6
9	0 0 0 1|1	0 0 1 0 1 0	8 10	20
	1 1 0 1|0	1 1 0 1 1	27	8.1
2	1 0 1 1	1 0 1 1	22	17,6
7 4	0 0 1 1 1 0	0 0 1 1 1 0	14	22,4
10	1 0 0|0 1	1 0 0 0	17	22.1
8	0 1 0|0 1	1 0 1 0 0 1	9	18,9
			Tổng	174,8
			Trung bình	17:48
			Tối đa	22,5

Cuối cùng, mỗi bit của nhiễm sắc thể mới đột biến với xác suất thấp. Đối với ví dụ này, chúng ta đặt xác suất đột biến là 0,001. Với tổng số 50 vị trí bit được chuyển, chúng tôi mong đợi 50·0,001 = 0,05 bit để thay đổi. Vì vậy, có khả năng là không có bit nào bị đột biến ở thế hệ thứ hai. Để minh họa, chúng tôi đã biến đổi một bit trong quần thể mới, điều này được thể hiện in đậm ở bảng 2.

Sau khi quá trình chọn lọc, lai ghép và đột biến hoàn tất, quần thể mới được kiểm tra bằng chức năng tập thể dục. Các giá trị thích nghi của thế hệ thứ hai này được liệt kê trong Bảng 2. Mặc dù rút ra kết luận từ một thử nghiệm duy nhất của thuật toán dựa trên xác suất là, “tốt nhất thì là một công việc kinh doanh rủi ro,” ví dụ này minh họa cách các thuật toán di truyền 'tiến hóa' theo hướng giải pháp ứng cử viên phù hợp hơn [1]. So sánh Bảng 2 với Bảng 1, chúng ta thấy rằng cả mức tối đa

thể lực và thể lực trung bình của dân số đã tăng lên chỉ sau một thế hệ.

2.2 Ví dụ: Số số 1 trong một chuỗi

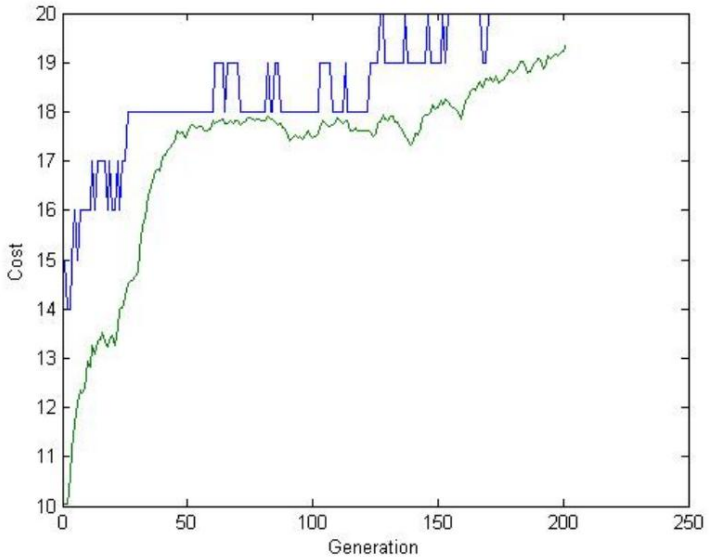
Ví dụ này điều chỉnh mã Haupt cho thuật toán di truyền nhị phân [3] cho máy tính đầu tiên
bài tập từ chư ơ ng 1 của sách giáo khoa Mitchell [7].

Trong ví dụ này, chúng ta sẽ lập trình một thuật toán di truyền hoàn chỉnh bằng MATLAB để tối đa hóa
mô phỏng số 1 trong một chuỗi bit có độ dài 20. Do đó nhiễm sắc thể của chúng ta sẽ là nhị phân
chuỗi có độ dài 20 và nhiễm sắc thể tối ư u mà chúng tôi đang tìm kiếm là

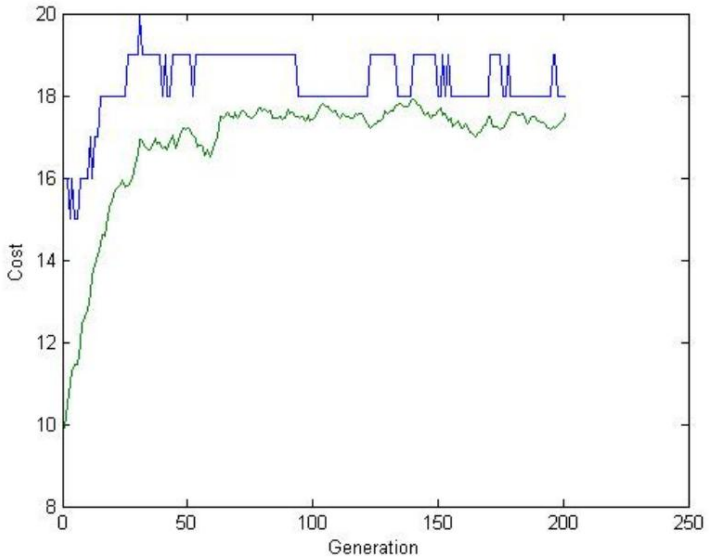
[11111111111111111111] .

Mã MATLAB hoàn chỉnh cho thuật toán này đư ợc liệt kê trong Phụ lục A. Vì thuật toán duy nhất
khả năng cho mỗi bit của nhiễm sắc thể là 1 hoặc 0, số lư ợng 1 trong nhiễm sắc thể
tư ơ ng đư ơ ng với tổng của tất cả các bit. Vì vậy, chúng tôi định nghĩa hàm thích ứng là
tổng số bit trong mỗi nhiễm sắc thể. Tiếp theo chúng ta xác định quy mô của dân số là 100
nhiễm sắc thể, tỷ lệ đột biến là 0,001 và số lần lặp lại (thế hệ)
là 200. Để đơ n giản, chúng ta đặt xác suất giao nhau là 1 và vị trí
chéo luôn luôn giống nhau.

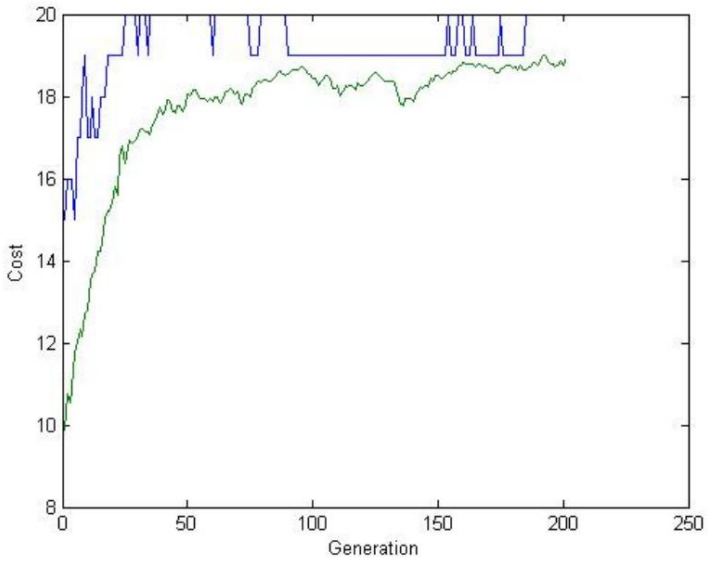
Hình 2, 3 và 4 thể hiện kết quả của ba lần chạy. Giải pháp tốt nhất đư ợc tìm thấy bởi cả hai lần chạy 1
và 3 là [11111111111111111111] . Giải pháp tốt nhất đư ợc tìm thấy trong lần chạy 2 là [11101111110111111111] .
Chúng ta có thể thấy từ lần chạy 2 (3) tại sao việc chạy toàn bộ thuật toán nhiều lần lại quan trọng
bởi vì lần chạy cụ thể này không tìm ra giải pháp tối ư u sau 200 thế hệ. Chạy 3
(4) có thể đã không tìm đư ợc giải pháp tối ư u nếu chúng ta quyết định lặp lại thuật toán ít hơ n
hơ n 200 lần. Những kết quả khác nhau này là do tính chất xác suất của một số bư ớc trong
thuật toán di truyền [5]. Nếu chúng ta gặp phải các bư ớc chạy như 2 và 3 (3, 4) trong tình huống thực tế,
sẽ là khôn ngoan nếu chúng ta tăng số lần lặp lại để đảm bảo rằng thuật toán
thực sự đã hội tụ về giải pháp tối ư u. Trong ứng dụng thực tế, ngư ời ta không biết



Hình 2: Lần chạy đầu tiên của thuật toán di truyền tối đa hóa số lượng số 1 trong chuỗi 20 bit. Đường cong màu xanh lam là mức độ thích nghi cao nhất và đường cong màu xanh lá cây là mức độ thích nghi trung bình. Giải pháp tốt nhất: [11111111111111111111] .



Hình 3: Lần chạy thứ hai của thuật toán di truyền tối đa hóa số lượng số 1 trong chuỗi 20 bit. Đường cong màu xanh lam là mức độ thích nghi cao nhất và đường cong màu xanh lá cây là mức độ thích nghi trung bình. Giải pháp tốt nhất: [11101111110111111111] .



Hình 4: Lần chạy thứ ba của thuật toán di truyền tối đa hóa số lượng số 1 trong chuỗi 20 bit. Đường cong màu xanh lam là mức độ thích nghi cao nhất và đường cong màu xanh lá cây là mức độ thích nghi trung bình. Giải pháp tốt nhất: [11111111111111111111] .

giải pháp tối ưu sẽ trông như thế nào, do đó, thông lệ là phải thực hiện nhiều lần chạy.

2.3 Ví dụ: Thuật toán di truyền liên tục

Ở đây chúng tôi trình bày một phong cách mới của thuật toán di truyền và gợi ý về các ứng dụng của nó trong địa hình. Ví dụ này điều chỉnh mã và ví dụ cho thuật toán di truyền liên tục từ Haupt's cuốn sách [3].

Bạn có thể nhận thấy rằng Ví dụ 2.1 chỉ cho phép nghiệm số nguyên, tức là có thể chấp nhận được trong trường hợp đó vì giá trị lớn nhất của hàm được đề cập là một số nguyên. Nhưng điều gì sẽ xảy ra nếu chúng ta cần thuật toán của mình để tìm kiếm trong một chuỗi giá trị liên tục? Trong trường hợp này nó làm cho hợp lý hơn khi biến mỗi nhiễm sắc thể thành một mảng các số thực (số dấu phẩy động) như trái ngược với một mảng chỉ có 0 và 1 [3]. Bằng cách này, độ chính xác của giải pháp chỉ bị giới hạn bởi máy tính, thay vì bởi biểu diễn nhị phân của các con số. Trong thực tế, Thuật toán di truyền liên tục này nhanh hơn thuật toán di truyền nhị phân vì nhiễm sắc thể không cần phải giải mã trước khi kiểm tra mức độ thích hợp của chúng bằng chức năng thích nghi.

vấn đề [3]. Hãy nhớ lại rằng nếu bài toán của chúng ta có kích thư ớc Npar thì chúng ta sẽ mã hóa từng kích thư ớc nhiễm sắc thể dư ới dạng mảng phần tử Npar

$$\text{nhiễm sắc thể} = [p1, p2, \ldots, pNpar]$$

trong đó mỗi pi là một giá trị cụ thể của i tham số, chỉ có điều lần này mỗi pi là số thực số thay vì một bit [2].

Giả sử chúng ta đang tìm kiếm điểm có độ cao thấp nhất trên địa hình (giả thuyết)-bản đồ ical trong đó kinh độ nằm trong khoảng từ 0 đến 10, vĩ độ nằm trong khoảng từ 0 đến 10 và độ cao của cảnh quan đư ợc cho bởi hàm

$$f(x, y) = x \sin(4x) + 1{,}1y \sin(2y).$$

Để giải quyết vấn đề này, chúng ta sẽ lập trình thuật toán di truyền liên tục bằng MATLAB. Các mã hoàn chỉnh cho thuật toán này đư ợc liệt kê trong Phụ lục B.

Chúng ta có thể thấy rằng f(x, y) sẽ là hàm thích nghi bởi vì chúng ta đang tìm kiếm một cái-giải pháp làm giảm thiểu độ cao. Vì f là hàm hai biến nên Npar = 2 và nhiễm sắc thể phải có dạng

$$\text{nhiễm sắc thể} = [x, y], \quad \text{với } 0 \le x \le 10 \text{ và } 0 \le y \le 10.$$

Chúng tôi xác định quy mô quần thể của chúng tôi là 12 nhiễm sắc thể, tỷ lệ đột biến là 0,2 và tỷ lệ số lần lặp là 100. Giống như trong thuật toán di truyền nhị phân, quần thể ban đầu đư ợc tạo ngẫu nhiên, ngoại trừ lần này các tham số có thể là bất kỳ số thực nào trong khoảng từ 0 và 10 thay vì chỉ 1 hoặc 0.

Với thuật toán này, chúng tôi chứng minh một cách tiếp cận hơ i khác để xây dựng các thế hệ tiếp theo hơ n chúng ta đã sử dụng trong các ví dụ trư ớc. Thay vì thay thế toàn bộ dân số có con mới, chúng tôi giữ cho một nửa dân số hiện tại khỏe mạnh hơ n và tạo ra

nửa còn lại của thế hệ mới thông qua lựa chọn và lai ghép. Bảng 3 và 4 cho thấy một quần thể ban đầu có thể có và một nửa trong số đó có thể sống sót sang thế hệ tiếp theo. Các toán tử lựa chọn chỉ chọn cha mẹ từ phần dân số được giữ lại này.

Bảng 3: Ví dụ về dân số ban đầu

x	y	Sự thích hợp f(x, y)
6.7874	6,9483	13.5468
7.5774	3,1710	-6.5696
7.4313	9,5022	-5.7656
3.9223	0,3445	0,3149
6.5548	4,3874	8.7209
1.7119	3,8156	5,0089
7.0605	7,6552	3.4901
0.3183	7,9520	-1.3994
2.7692	1,8687	-3.9137
0.4617	4,8976	-1.5065
0.9713	4,4559	1.7482
8.2346	6.4631	10.7287

Bảng 4: Quần thể sống sót sau tỷ lệ lựa chọn 50%

Xếp hạng x	y	Sự thích hợp f(x, y)
7.5774	3,1710	-6.5696
1 2 7.4313	9,5022	-5.7656
2.7692	1,8687	-3.9137
3 4 0.4617 5	4,8976	-1.5065
0.3183 6	7,9520	-1.3994
3.9223	0,3445	0,3149

Lưu ý rằng hàm thích nghi nhận cả giá trị dương và âm, vì vậy chúng ta không thể trực tiếp sử dụng tổng giá trị thích hợp của nhiễm sắc thể để xác định xác suất của ai sẽ được chọn để sao chép (như chúng tôi đã làm trong Ví dụ 2.1). Thay vào đó chúng tôi sử dụng trọng số xếp hạng, dựa trên thứ hạng được thể hiện trong Bảng 4. Xác suất nhiễm sắc thể ở n địa điểm sẽ là cha mẹ được cho bởi phương trình

$$P_n = \frac{N_{keep}\quad n+1}{N_{keep}} = \frac{6\quad n+1}{1+2+3+4+5+6} = \frac{7\quad n}{21},$$

vậy xác suất để nhiễm sắc thể ở vị trí đầu tiên được chọn là

$$P(C1) = \frac{6}{1+2+3+4+5+6} \cdot \frac{1+1}{} = \frac{6}{21},$$

và xác suất để nhiễm sắc thể ở vị trí thứ sáu được chọn là

$$P(C6) = \frac{6}{1+2+3+4+5+6} \cdot \frac{6+1}{} = \frac{1}{21}.$$

Hãy nhớ lại rằng mỗi lần giao phối tạo ra 2 con cái, vì vậy chúng ta cần 3 cặp nhiễm sắc thể bố mẹ để sinh ra đủ số lượng con cháu để lấp đầy phần còn lại của thế hệ tiếp theo.

Vì nhiễm sắc thể của chúng ta không còn là các chuỗi bit nữa nên chúng ta cần xem lại cách thức trao đổi chéo và các toán tử đột biến hoạt động. Có một số phương pháp để đạt được điều này; ở đây chúng tôi sẽ sử dụng Phương pháp Haupt [3]. Khi chúng ta có hai nhiễm sắc thể bố mẹ m = [xm, ym] và d = [xd, yd], đầu tiên chúng tôi chọn ngẫu nhiên một trong các tham số làm điểm giao nhau. Để minh họa, giả sử x là điểm giao nhau của m và d cụ thể. Sau đó chúng tôi giới thiệu một giá trị ngẫu nhiên giữa 0 và 1 được biểu thị bằng β và giá trị x ở con cái là

$$\text{xnew1} = (1 - \beta)\text{xm} + \beta\,\text{xd}$$
$$\text{xnew2} = (1 - \beta)\text{xd} + \beta\,\text{xm}.$$

Tham số còn lại (y trong trường hợp này) được kế thừa trực tiếp từ mỗi cha mẹ, do đó con cái đã hoàn thành là

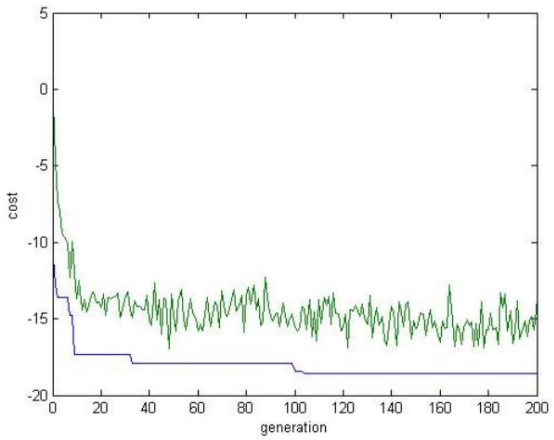
$$\text{con cháu1} = [\text{xnew1}, \text{ym}]$$
$$\text{con cháu2} = [\text{xnew2}, \text{yd}].$$

Nếu chúng ta giả sử rằng nhiễm sắc thể1 = [7.5774 , 3.1710] và nhiễm sắc thể3 = [2.7692 , 1.8687] được chọn làm cặp bố mẹ. Sau đó, với sự giao nhau tại x và giá trị β là 0,3463, con cháu sẽ là

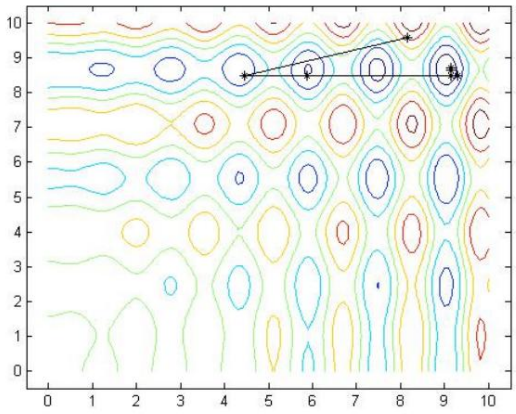
$$\text{con1} = [(1 - 0{,}3463) \cdot 7{,}5774 + 0{,}3463 \cdot 2{,}7692,\ 3{,}1710] = [5{,}9123,\ 3{,}1710]$$

con2 = [(1 0,3463) · 2,7692 + 0,3463 · 7,5774, 1,8687] = [4,4343, 1,8687] .

Giống như toán tử chéo, có nhiều phương pháp khác nhau để điều chỉnh toán tử sang thuật toán di truyền liên tục. Đối với ví dụ này, khi một tham số trong một nhiễm sắc thể bị đột biến, giá trị của nó được thay thế bằng một số ngẫu nhiên trong khoảng từ 0 đến 10.



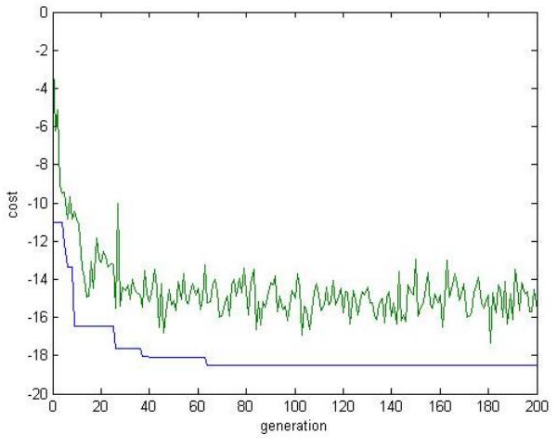
(a) Đường cong màu xanh là độ cao thấp nhất, đường cong màu xanh lá cây là độ cao trung bình.



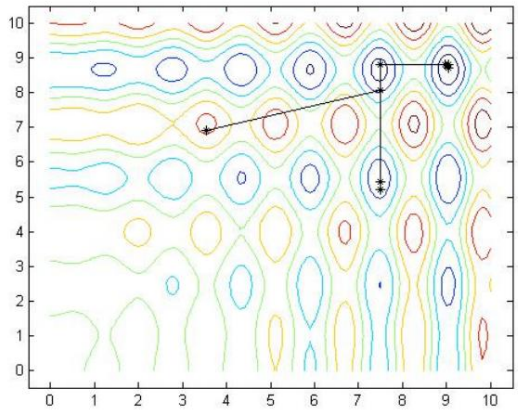
(b) Bản đồ đường viền hiển thị 'đường đi' của nhiễm sắc thể phù hợp nhất từ thế hệ này sang thế hệ tiếp theo.

Hình 5: Lần chạy đầu tiên của thuật toán di truyền liên tục tìm điểm có độ cao thấp nhất của hàm f(x, y) = x sin(4x) + 1,1y sin(2y) trong vùng 0 ≤ x ≤ 10 và 0 ≤ y 10.
Với độ cao -18,5519, giải pháp tốt nhất là [9,0449, 8,6643].

Hình 5, 6 và 7 thể hiện kết quả của ba lần chạy. Chúng ta có thể thấy trong cả ba lần chạy rằng thuật toán tìm thấy thời gian tối thiểu trước khi đạt đến thế hệ thứ 200. Trong điều đặc biệt này trường hợp, số thế hệ quá nhiều không phải là vấn đề vì thuật toán đơn giản và hiệu quả của nó không bị ảnh hưởng đáng kể. Lưu ý rằng bởi vì các thành viên tốt nhất của một nhóm nhất định thế hệ tồn tại sang thế hệ tiếp theo, giá trị tốt nhất của f luôn được cải thiện hoặc vẫn như nhau. Chúng tôi cũng thấy rằng giải pháp cuối cùng của mỗi lần chạy hơi khác so với những giải pháp khác. Điều này không có gì đáng ngạc nhiên vì các tham số của chúng tôi là các biến liên tục và do đó có vô số giá trị có thể có tùy ý gần với mức tối thiểu thực tế.

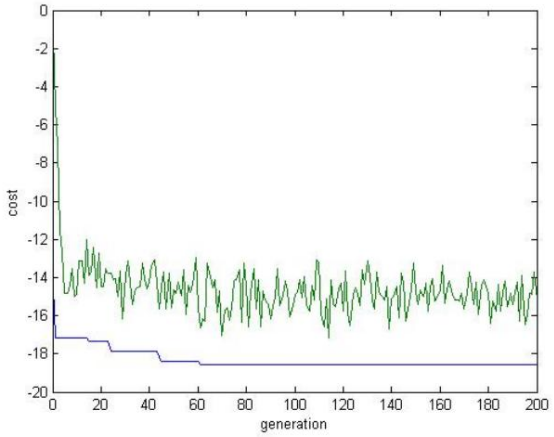


(a) Đường cong màu xanh là độ cao thấp nhất, đường cong màu xanh lá cây là độ cao trung bình.

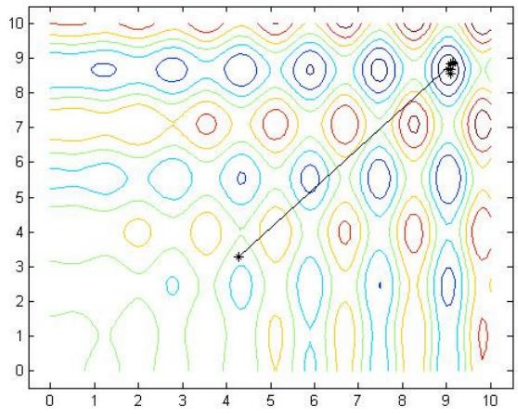


(b) Bản đồ đường viền hiển thị 'đường dẫn' của nhiễm sắc thể phù hợp nhất từ thế hệ này sang thế hệ tiếp theo.

Hình 6: Lần chạy thứ hai của thuật toán di truyền liên tục tìm điểm có độ cao thấp nhất của hàm f(x, y) = x sin(4x) + 1,1y sin(2y) trong vùng 0 ≤ x ≤ 10 và 0 ≤ y ≤ 10. Với độ cao -18,5227, nghiệm tốt nhất là [9,0386, 8,709].



(a) Đường cong màu xanh là độ cao thấp nhất, đường cong màu xanh lá cây là độ cao trung bình.



(b) Bản đồ đường viền hiển thị 'đường dẫn' của nhiễm sắc thể phù hợp nhất từ thế hệ này sang thế hệ tiếp theo.

Hình 7: Lần thứ ba của thuật toán di truyền liên tục tìm điểm có độ cao thấp nhất của hàm f(x, y) = x sin(4x) + 1,1y sin(2y) trong vùng 0 ≤ x ≤ 10 và 0 ≤ y 10.
Với độ cao -18,5455, giải pháp tốt nhất là [9,0327, 8,6865].

3 Bối cảnh lịch sử

Ngay từ những năm 1950, các nhà khoa học đã nghiên cứu trí tuệ nhân tạo và công nghệ tiến hóa.
giả định, cố gắng viết các chư ơ ng trình máy tính có thể mô phỏng các quá trình xảy ra trong
thế giới tự nhiên. Năm 1953, Nils Barricelli đư ợc mời tới Princeton để nghiên cứu về trí tuệ nhân tạo
[9]. Anh ấy đã sử dụng máy tính kỹ thuật số mới đư ợc phát minh gần đây để viết phần mềm mô phỏng tự nhiên
sinh sản và đột biến. Mục tiêu của Barricelli không phải là giải quyết các vấn đề tối ư u hóa hay
mô phỏng sự tiến hóa sinh học mà đúng hơ n là tạo ra sự sống nhân tạo. Ông đã tạo ra gen đầu tiên
phần mềm thuật toán, và tác phẩm của ông đã đư ợc xuất bản bằng tiếng Ý vào năm 1954 [9]. Công việc của Barricelli là
tiếp theo vào năm 1957 bởi Alexander Fraser, một nhà sinh vật học đến từ London. Ông đã có ý tư ởng tạo ra
một mô hình tiến hóa trên máy tính, vì việc quan sát trực tiếp nó sẽ cần hàng triệu năm.
Ông là ngư ời đầu tiên chỉ sử dụng chư ơ ng trình máy tính để nghiên cứu sự tiến hóa [9]. Nhiều nhà sinh học
theo bư ớc chân của ông vào cuối những năm 1950 và 1960.

Trong cuốn sách của mình, Mitchell nói rằng John Holland đã phát minh ra thuật toán di truyền vào những năm 1960,
hoặc ít nhất là phiên bản cụ thể đư ợc biết đến ngày nay với tiêu đề cụ thể đó [7]. của Hà Lan
phiên bản liên quan đến sự mô phỏng về “sự sống sót của kẻ mạnh nhất” theo thuyết Darwin, cũng như các quá trình
chéo, tái tổ hợp, đột biến và đảo ngư ợc xảy ra trong di truyền [7]. Dân số này-
phư ơ ng pháp dựa trên là một sự đổi mới lớn. Các thuật toán di truyền trư ớc đây chỉ sử dụng đột biến làm
động lực của sự tiến hóa [7] Holland là giáo sư tâm lý học, khoa học máy tính và điện
kỹ thuật tại Đại học Michigan. Anh ta bị thúc đẩy bởi sự theo đuổi để hiểu làm thế nào
“các hệ thống thích ứng với môi trư ờng xung quanh” [9]. Sau nhiều năm hợp tác với sinh viên và
đồng nghiệp, ông đã xuất bản cuốn sách nổi tiếng Thích ứng trong các hệ thống tự nhiên và nhân tạo ở
1975. Trong cuốn sách này, Holland đã trình bày các thuật toán di truyền như một “sự trừu tư ợng của sinh học
tiến hóa và đư a ra khung lý thuyết cho sự thích ứng” theo thuật toán di truyền [7].
Cuốn sách này là cuốn sách đầu tiên đề xuất nền tảng lý thuyết cho sự phát triển tính toán,
và nó vẫn là nền tảng của mọi công trình lý thuyết về thuật toán di truyền cho đến gần đây [7].
Nó đã trở thành một tác phẩm kinh điển vì nó thể hiện tính toán học đằng sau quá trình tiến hóa [9].
Cùng năm đó, một trong những nghiên cứu sinh tiến sĩ của Hà Lan, Kenneth De Jong, đã trình bày luận điểm đầu tiên

nghiên cứu toàn diện về việc sử dụng thuật toán di truyền để giải các bài toán tối ưu hóa như
Luận án tiến sĩ. Công trình nghiên cứu của ông kỹ lưỡng đến mức trong nhiều năm, bất kỳ tài liệu nào về di truyền học đều
các thuật toán không bao gồm các ví dụ chuẩn của anh ấy được coi là "không đủ" [9].

Nghiên cứu về thuật toán di truyền tăng nhanh trong những năm 1970 và 1980. Điều này một phần
nhờ những tiến bộ trong công nghệ. Các nhà khoa học máy tính cũng bắt đầu nhận ra những hạn chế của
phương pháp lập trình thông thường và tối ưu hóa truyền thống để giải quyết các vấn đề phức tạp
lem. Các nhà nghiên cứu phát hiện ra rằng thuật toán di truyền là một cách để tìm ra giải pháp cho các vấn đề
mà các phương pháp khác không thể giải quyết được. Thuật toán di truyền có thể kiểm tra đồng thời nhiều điểm
từ khắp không gian giải pháp, tối ưu hóa với các tham số rời rạc hoặc liên tục, hỗ trợ
video một số thông số tối ưu thay vì một giải pháp duy nhất và làm việc với nhiều giải pháp khác nhau
các loại dữ liệu [2] Những ưu điểm này cho phép các thuật toán di truyền "tạo ra những kết quả đáng kinh ngạc".
khi các phương pháp tối ưu hóa truyền thống thất bại thảm hại" [2].

Khi nói đến các phương pháp tối ưu hóa "truyền thống", chúng tôi đang đề cập đến ba loại chính:
dựa trên phép tính, tìm kiếm toàn diện và ngẫu nhiên [1]. Phương pháp tối ưu hóa dựa trên tính toán
chia làm hai loại: trực tiếp và gián tiếp. Phương pháp trực tiếp "nhảy vào" mục tiêu
hoạt động và tuân theo hướng của gradient hướng tới mức tối đa hoặc tối thiểu cục bộ
giá trị. Đây còn được gọi là phương pháp "leo đồi" hoặc "đi lên dốc" [1]. Các
phương pháp gián tiếp lấy gradient của hàm mục tiêu, đặt nó bằng 0, sau đó
giải tập hợp các phương trình thu được [1]. Mặc dù các kỹ thuật dựa trên tính toán này có
được nghiên cứu rộng rãi và cải tiến nhưng chúng vẫn còn tồn tại những vấn đề chưa thể khắc phục. Đầu tiên, họ
chỉ tìm kiếm tối ưu cục bộ, điều này khiến chúng trở nên vô dụng nếu chúng ta không biết vùng lân cận
về mức tối ưu toàn cục hoặc liệu có mức tối ưu cục bộ nào khác ở gần đó hay không (và chúng ta thường không biết).
Thứ hai, những phương pháp này yêu cầu sự tồn tại của đạo hàm và điều này hầu như không bao giờ là
trường hợp ứng dụng thực tế.

Các thuật toán tìm kiếm toàn diện thực hiện chính xác điều đó—một tìm kiếm toàn diện. Những thuật toán này
các nhịp điệu yêu cầu một "không gian tìm kiếm hữu hạn hoặc một không gian tìm kiếm vô hạn rời rạc" của các giá trị có thể
cho hàm mục tiêu [1]. Sau đó, họ kiểm tra từng giá trị một để tìm ra giá trị

tối đa hoặc tối thiểu. Mặc dù phương pháp này đơn giản và do đó “hấp dẫn” nhưng nó lại kém hiệu quả nhất.
hiệu quả của tất cả các thuật toán tối ưu hóa [1]. Trong các bài toán thực tế, không gian tìm kiếm quá
rộng lớn để kiểm tra mọi khả năng “mỗi lần một khả năng và vẫn có cơ hội sử dụng [kết quả]
thông tin nhằm đạt được mục đích thực tế nào đó” [1].

Các thuật toán tìm kiếm ngẫu nhiên ngày càng trở nên phổ biến khi mọi người nhận ra những thiếu sót
các thuật toán tìm kiếm toàn diện và dựa trên phép tính. Kiểu thuật toán này ngẫu nhiên
chọn một số mẫu đại diện từ không gian tìm kiếm và tìm giá trị tối ưu trong
việc lấy mẫu đó. Mặc dù nhanh hơn tìm kiếm toàn diện nhưng phương pháp này “có thể thực hiện được
không tốt hơn” so với một tìm kiếm toàn diện [1]. Sử dụng loại thuật toán này có nghĩa là chúng ta rời khỏi
tùy thuộc vào cơ hội liệu chúng ta sẽ ở gần giải pháp tối ưu hay cách xa giải pháp tối ưu hàng dặm.
Nó.

Thuật toán di truyền có nhiều ưu điểm so với các phương pháp truyền thống này. Không giống như phép tính-
dựa trên các phương pháp như leo đồi, thuật toán di truyền phát triển từ quần thể ứng cử viên
giải pháp thay vì một giá trị duy nhất [1]. Điều này làm giảm đáng kể khả năng tìm thấy một địa phương
tối ưu thay vì tối ưu toàn cục. Thuật toán di truyền không yêu cầu thêm thông tin
(như đạo hàm) không liên quan đến giá trị của chính các giải pháp khả thi.
Cơ chế duy nhất hướng dẫn tìm kiếm của họ là giá trị số phù hợp của ứng viên
giải pháp, dựa trên định nghĩa về sự phù hợp của người sáng tạo [5]. Điều này cho phép chúng hoạt động khi
không gian tìm kiếm ồn ào, phi tuyến tính và đạo hàm thậm chí không tồn tại. Điều này cũng khiến họ
có thể áp dụng trong nhiều tình huống hơn các thuật toán truyền thống và chúng có thể được điều chỉnh theo
mỗi tình huống dựa trên việc độ chính xác hay hiệu quả là quan trọng hơn.

4 vấn đề thực tế

Trong phần này chúng tôi trình bày hai bài toán tối ưu hóa cổ điển bằng thuật toán di truyền.
Chúng tôi sẽ lập trình các thuật toán này bằng MATLAB.

4.1 Bài toán ngư ời bán hàng du lịch

Đây có lẽ là phư ơ ng pháp tối ư u hóa tổ hợp nổi tiếng, thực tế và đư ợc nghiên cứu rộng rãi nhất.
vấn đề (theo tổ hợp, chúng tôi muốn nói nó liên quan đến hoán vị—sắp xếp một số lư ợng nhất định các
các mặt hàng theo thứ tự khác nhau) [9]. Nó có ứng dụng trong “ngư ời máy, khoan bảng mạch, hàn,
sản xuất, vận tải và nhiều lĩnh vực khác” [9]. Ví dụ: “một bảng mạch
có thể có hàng chục nghìn lỗ và cần phải lập trình một mũi khoan để thăm từng lỗ.
những lỗ hổng đó trong khi giảm thiểu một số chức năng chi phí (ví dụ như thời gian hoặc năng lư ợng)” [9].

Trong bài toán này, chúng ta giả sử rằng có một nhân viên bán hàng du lịch cần đến thăm n thành phố
bằng con đư ờng ngắn nhất có thể. Anh ta đến thăm mỗi thành phố đúng một lần, sau đó quay lại thành phố đó
nơ i anh ấy bắt đầu. Do đó, giải pháp ứng cử viên sẽ là danh sách tất cả các thành phố theo thứ tự
rằng anh ấy đến thăm họ [3]. Chúng ta ký hiệu các thành phố là thành phố 1, thành phố 2, thành phố 3, . . . , thành phố n; thành phố ở đâu
1 là điểm khởi đầu của anh ấy. Như vậy các nhiễm sắc thể dùng cho thuật toán di truyền sẽ khác nhau
hoán vị của các số nguyên từ 1 đến n.

Có rất nhiều biến thể của bài toán ngư ời bán hàng du lịch, như ng với mục đích của chúng ta, chúng ta
đư a ra các giả định sau. Giả sử khoảng cách d từ thành phố ci đến thành phố cj là
d(ci , cj) = d(cj , ci) với mọi i [1, n] và j [1, n]. Chúng ta có thể thấy trong một số trư ờng hợp thực tế
điều này sẽ không đúng: đư ờng cao tốc ở một hư ớng có thể dài hơ n đư ờng cao tốc một chút
đi theo hư ớng ngư ợc lại, hoặc lái xe lên dốc có thể tốn kém hơ n so với xuống dốc (đó là
thông thư ờng chi phí đi lại đư ợc tính vào cái mà chúng ta gọi là “khoảng cách”). Những trư ờng hợp đó
đư ợc gọi là bài toán ngư ời bán hàng du lịch “không đối xứng” [9]. Vì ngư ời bán hàng kết thúc ở thành phố
nơ i anh ấy bắt đầu, đây là một vấn đề nhân viên bán hàng du lịch “đóng”. Trong biến thể “mở”,
ngư ời bán hàng đến thăm tất cả các thành phố đúng một lần, vì vậy anh ta không kết thúc hành trình nơ i mình đã bắt đầu [9].

4.1.1 Sửa đổi chéo và đột biến cho các vấn đề hoán vị

Dựa trên cách chúng tôi mã hóa các giải pháp ứng viên thành nhiễm sắc thể, các toán tử của
lai ghép và đột biến như chúng ta đã định nghĩa chúng cho thuật toán di truyền nhị phân lúc đầu
của bài viết này sẽ không hiệu quả vì mỗi thành phố chỉ cần đư ợc đại diện một lần và chỉ một lần [3].

Để biết lý do tại sao lại như vậy, hãy xem xét hai nhiễm sắc thể bố mẹ có chiều dài 5 và con cái của chúng sau khi chéo bình thường:

Cha mẹ	Con cái bình thường chéo (bị lỗi)
3 5 1 2 4 3 5|1 2 4 3 5 5 3 2	
1 4 5 3 2 1 4 1|2 4|5 3 2	

Những con này không phải là hoán vị hợp lệ của các số nguyên từ 1 đến 5 vì chúng là thiếu một số số và lặp lại những số khác. Để vượt qua trở ngại này, chúng tôi sẽ sử dụng một kỹ thuật được gọi là chéo "chu kỳ" trong thuật toán di truyền của chúng tôi [3]. Có một số cách khác để sửa đổi chéo để phù hợp với hoán vị, được thảo luận trong Goldberg [1] và Haupt [3]. Sự trao đổi chéo chu kỳ được minh họa bằng các nhiễm sắc thể có chiều dài 6 trong Bảng 5. Đầu tiên chúng ta

Bảng 5: Ví dụ về chuyển đổi chu kỳ

Cha mẹ	Con cái (bước 1)	Con cái (bước 2)	Con cái (bước 3)	Con cháu (bước 4)
4 1 5|3|2 6	4 1 5 2|2|6	4|1|5 2 1 6	|4|4 5 2 1 6 |	3 4 5 2 1 6
3 4 6|2|1 5	3 4 6 3|1|5	3|4|6 3 2 5	3|1 6 3 2 5	4 1 6 3 2 5

chọn một vị trí ngẫu nhiên trên chiều dài của nhiễm sắc thể. Hai nhiễm sắc thể bố mẹ trao đổi số nguyên tại vị trí này (được đánh dấu bằng các vạch trong Bảng 5) để tạo ra con cái. Trừ khi các số nguyên được hoán đổi có cùng giá trị, mỗi con có một số nguyên trùng lặp. Sau đó chúng tôi chuyển đổi số nguyên trùng lặp ở thế hệ con đầu tiên với bất kỳ số nguyên nào ở cùng một vị trí ở thế hệ con thứ hai. Điều này có nghĩa là bây giờ có một bản sao của một số nguyên khác, vì vậy quá trình lặp lại cho đến khi không còn bản sao nào ở con đầu tiên (hoặc con thứ hai) [3]. Bây giờ mỗi con có chính xác một trong mọi số nguyên, do đó cả hai đều là hoán vị hợp lệ.

Toán tử đột biến sửa đổi chọn ngẫu nhiên hai số nguyên trong nhiễm sắc thể từ thế hệ mới và hoán đổi chúng. Toán tử này vẫn thường xảy ra với xác suất thấp.

4.1.2 Sử dụng thuật toán di truyền

Trong phần này chúng ta xây dựng thuật toán di truyền để giải bài toán nhân viên bán hàng du lịch. ĐẾN làm như vậy, chúng tôi điều chỉnh và cải tiến mã Haupt cho thuật toán di truyền hoán vị và cho hàm thích hợp của bài toán nhân viên bán hàng du lịch [3]. Mã MATLAB hoàn chỉnh cho thuật toán này được liệt kê trong Phụ lục C.

Hãy để có 20 thành phố. Để biểu diễn chúng, chúng ta đặt ngẫu nhiên 20 điểm trên mặt phẳng xy trong hình vuông 1 × 1 giữa (0, 0) và (1, 1). Trước đó, chúng ta đã định nghĩa nhiễm sắc thể của mình là hoán vị của các số nguyên từ 1 đến 20.

Giả sử n thành phố được liệt kê theo thứ tự c1 c2 · · · cn theo một thời gian nhất định mosome. Vì chúng ta đang cố gắng giảm thiểu tổng khoảng cách mà người bán hàng phải đi, nên hàm thích nghi sẽ là tổng khoảng cách D:

$$D = \sum_{k=1}^{N} d(c_k, c_{k+1})$$

trong đó thành phố n + 1 = thành phố 1 (thành phố xuất phát) [3]. Nếu ta gọi (xi , yi) biểu thị tọa độ của thành phố ci , thì hàm thích nghi sẽ trở thành

$$D = \sum_{k=1}^{N} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2}.$$

Tiếp theo, chúng tôi xác định quy mô dân số là 20. Với thuật toán này, chúng tôi sử dụng cùng một phương pháp xây dựng các thế hệ tiếp theo mà chúng tôi đã sử dụng trong Ví dụ 2.3. Hơn là thay vì thay thế toàn bộ quần thể bằng con cái mới, chúng ta giữ lại một nửa dân số hiện tại tốt hơn dân số, và tạo ra nửa còn lại của thế hệ mới thông qua chọn lọc và lai ghép.
Hãy nhớ lại rằng toán tử lựa chọn chỉ chọn từ phần tổng thể đã giữ. Chúng tôi đã sửa đổi các toán tử chéo và đột biến ở trên (4.1.1). Trong thuật toán này, toán tử đột biến tác động lên từng thành viên của thế hệ mới với xác suất 0,05.

Điểm mà thuật toán này sai lệch nhiều nhất so với phiên bản gốc của Haupt là ở

tính toán phân bố xác suất cho toán tử lựa chọn, đư ợc hiển thị trong Liệt kê 4.1.

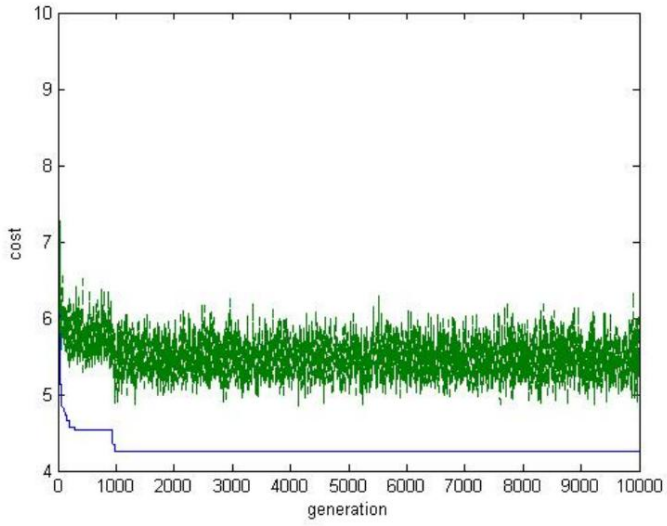
```
với ii =2: giữ tỷ lệ
       cư ợc =[ tỷ lệ cư ợc ii * số một (1  , ii )];
kết thúc
Gật đầu = chiều dài ( tỷ lệ cư ợc );
tỷ lệ cư ợc = giữ - tỷ lệ cư ợc +1;
```

Liệt kê 4.1: Xác định xác suất cho toán tử lựa chọn dựa trên xếp hạng mức độ phù hợp của 10 nhiễm sắc thể hàng đầu.

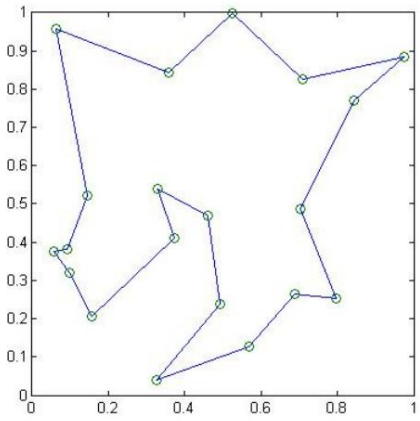
Những gì chúng tôi đã làm ở đây để sửa đổi phiên bản của Haupt là thêm dòng cuối cùng, điều này có hiệu quả đảo ngư ợc sự phân bố xác suất của nhóm nhiễm sắc thể đư ợc chọn từ sinh sản. Trư ớc điểm này trong mã, thế hệ nhiễm sắc thể hiện tại đư ợc sắp xếp từ thể lực cao nhất đến thể lực thấp nhất (từ 1 đến 20) và chỉ một nửa khỏe mạnh hơ n (10 nhiễm sắc thể) bị giữ. Tỷ lệ cư ợc thay đổi trong Liệt kê 4.1 là một vectơ chứa các số nguyên từ 1 đến 10, với mư ời trư ờng hợp của số nguyên 1, chín trư ờng hợp của số nguyên 2, xuống còn một trư ờng hợp của số nguyên 10. Một nhiễm sắc thể sau đó đư ợc chọn làm cha mẹ bằng cách chọn một số nguyên tại ngẫu nhiên từ tỷ lệ vectơ . Do đó chúng ta thấy rằng xác suất của nhiễm sắc thể phù hợp nhất đư ợc chọn là $\frac{10}{55} = \frac{}{2\ 11}$, và xác suất nhiễm sắc thể ít phù hợp nhất đư ợc chọn là $\frac{}{1\ 55}$. Nếu không có dòng mã mới mà chúng tôi đã thêm vào, tỷ lệ cư ợc sẽ có mư ời số 10, chín số 9, v.v. sẽ mang lại cho nhiễm sắc thể ít phù hợp nhất khả năng trở thành cha mẹ cao nhất và ngư ợc lại.

Cuối cùng, chúng tôi xác định số lần lặp là 10.000.

Các hình 8, 9, 10, 11 và 12 thể hiện năm trong số nhiều lần chạy. Vì chúng ta không biết tiên nghiệm điều gì giải pháp tối ư u là, điều quan trọng là phải chạy toàn bộ thuật toán nhiều lần cho đến khi kết quả tốt nhất xuất hiện lại một số lần thỏa đáng. Chúng ta có thể thấy từ Hình 9 và Hình 11 khiến chúng ta có thể bị lừa khi nghĩ rằng 4,1816 là khoảng cách ngắn nhất có thể nếu chúng ta không chạy thuật toán đủ số lần để kết quả 4.1211 xuất hiện. May mắn thay, 4.1211 xuất hiện chỉ sau ba lần chạy (Hình 10). Một số lần chạy nữa đã xác nhận rằng không có

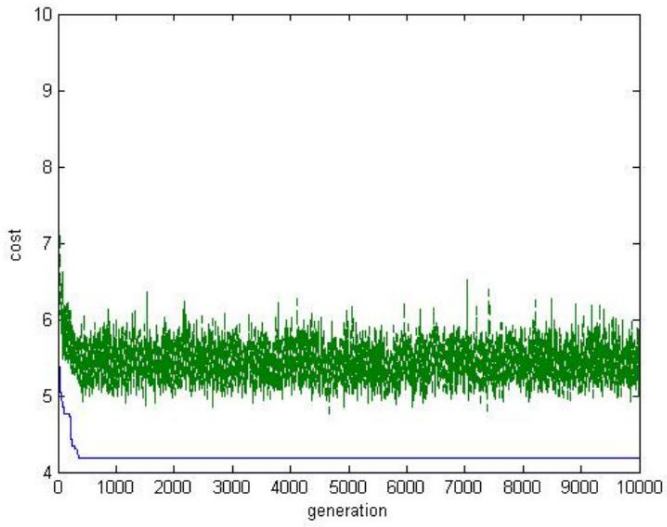


(a) Đường cong màu xanh là khoảng cách ngắn nhất, đường cong màu xanh lá cây là khoảng cách trung bình.

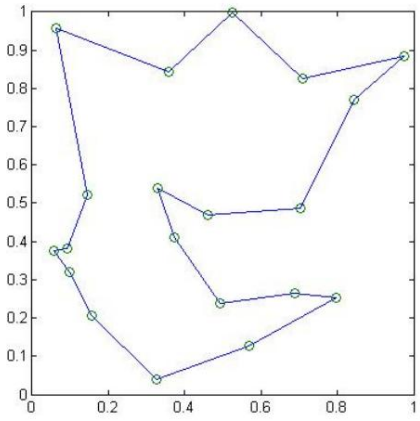


(b) Đồ thị của tuyến đường.

Hình 8: Lần chạy đầu tiên của thuật toán di truyền giảm thiểu khoảng cách di chuyển theo một vòng khép kín tới 20 thành phố. Với khoảng cách 4,2547, giải pháp tốt nhất là [13, 15, 4, 18, 11, 17, 10, 14, 1, 3, 19, 12, 5, 20, 8, 2, 9, 7, 16, 6].

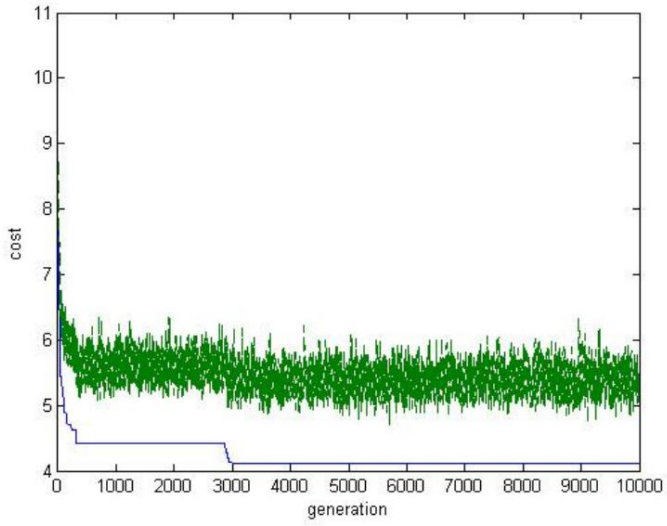


(a) Đường cong màu xanh là khoảng cách ngắn nhất, đường cong màu xanh lá cây là khoảng cách trung bình.

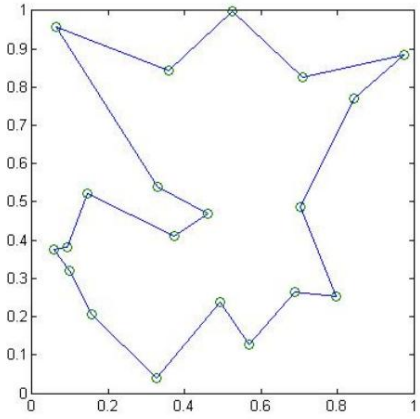


(b) Đồ thị của tuyến đường.

Hình 9: Lần chạy thứ hai của thuật toán di truyền giảm thiểu khoảng cách di chuyển theo một vòng khép kín tới 20 thành phố. Với khoảng cách 4,1816, giải pháp tốt nhất là [4, 18, 11, 17, 10, 14, 1, 3, 19, 2, 9, 16, 7, 8, 12, 5, 20, 6, 13, 15].

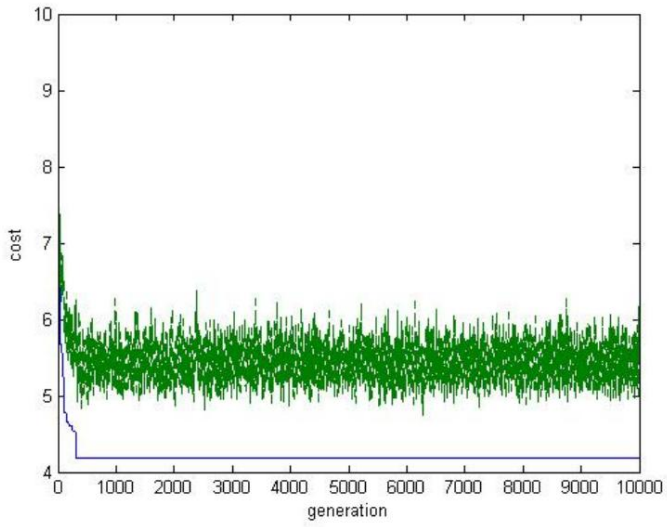


(a) Đường cong màu xanh là khoảng cách ngắn nhất, đường cong màu xanh lá cây là khoảng cách trung bình.

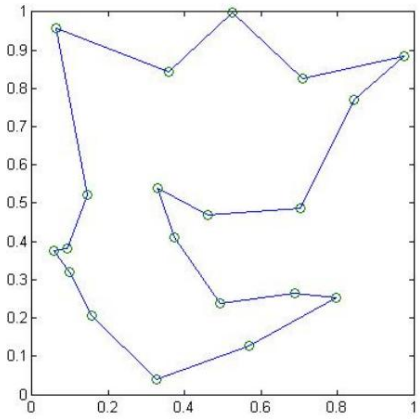


(b) Đồ thị của tuyến đường.

Hình 10: Lần chạy thứ ba của thuật toán di truyền giảm thiểu khoảng cách di chuyển theo một vòng khép kín tới 20 thành phố. Với khoảng cách 4.1211, giải pháp tốt nhất là [7, 9, 8, 2, 19, 3, 1, 14, 10, 12, 20, 5, 17, 11, 18, 4, 15, 13, 6, 16].

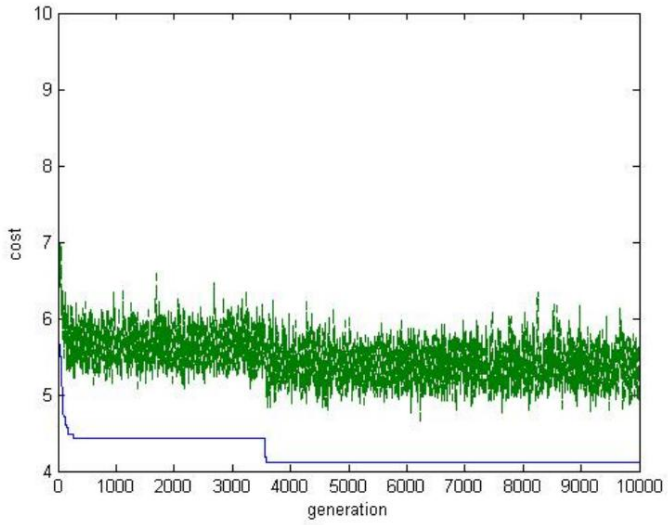


(a) Đường cong màu xanh là khoảng cách ngắn nhất, đường cong màu xanh lá cây là khoảng cách trung bình.

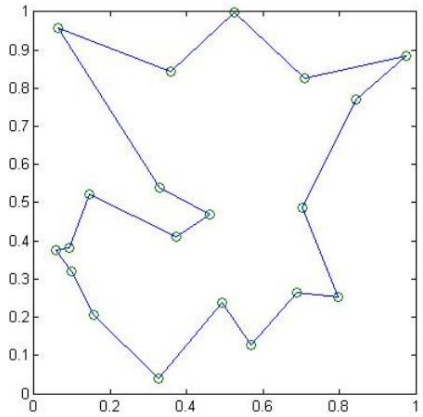


(b) Đồ thị của tuyến đường.

Hình 11: Lần chạy thứ tư của thuật toán di truyền giảm thiểu khoảng cách di chuyển theo một vòng khép kín tới 20 thành phố. Với khoảng cách 4,1816, giải pháp tốt nhất là [10, 14, 1, 3, 19, 2, 9, 16, 7, 8, 12, 5, 20, 6, 13, 15, 4, 18, 11, 17].



(a) Đường cong màu xanh là khoảng cách ngắn nhất, đường cong màu xanh lá cây là khoảng cách trung bình.



(b) Đồ thị của tuyến đường.

Hình 12: Lần chạy thứ mười một của thuật toán di truyền giảm thiểu khoảng cách di chuyển theo một vòng khép kín tới 20 thành phố. Với khoảng cách 4.1211, giải pháp tốt nhất là [20, 12, 10, 14, 1, 3, 19, 2, 8, 9, 7, 16, 6, 13, 15, 4, 18, 11, 17, 5].

khoảng cách ngắn hơn 4,1211 nên nghiệm tối ưu là

[7, 9, 8, 2, 19, 3, 1, 14, 10, 12, 20, 5, 17, 11, 18, 4, 15, 13, 6, 16] .

Lượt chạy 11 (Hình 12) là một trong những lượt chạy đã xác nhận điều này. Lưu ý rằng giải pháp tối ưu chạy 11 được tìm thấy tương đương với chạy 3 mặc dù nhiễm sắc thể là một hoán vị. Nhiễm sắc thể này bắt đầu ở một thành phố khác, nhưng điều này không ảnh hưởng đến khoảng cách vì con đường của người bán hàng là một vòng khép kín. Nhiễm sắc thể này cũng liệt kê các thành phố ở thứ tự ngược lại với lần chạy 3, nhưng hãy nhớ lại rằng

d(ci , cj) = d(cj , ci) cho các thành phố ci , cj

vì vậy điều này cũng không ảnh hưởng đến khoảng cách.

4.1.3 Sử dụng thuật toán truyền thống

Tại sao bài toán người bán hàng du lịch lại khó đến vậy trước khi thuật toán di truyền được phát minh? Chúng ta hãy xem xét việc sử dụng một thuật toán tìm kiếm đầy đủ để giải quyết nó. Lưu ý rằng vì người bán hàng di chuyển theo một vòng khép kín và hướng di chuyển giữa các thành phố không quan trọng (thông tư miễn phí hoán vị), tổng số nghiệm có thể là

$$S = \frac{(n-1)!\ 2}{} = \frac{19!}{2} = 60, 822, 550, 204, 416, 000 .$$

Con số này quá lớn để kiểm tra từng ứng viên ngay cả với các giá trị n nhỏ hơn.
hơn 20. Lượng thời gian cần thiết để một thuật toán tìm kiếm toàn diện tìm thấy
giải pháp tốt nhất sẽ vô hiệu hóa hoàn toàn tính hữu ích của bất kỳ câu trả lời nào nó tìm thấy.

4.2 Vấn đề về chiếc ba lô

Ở đây chúng tôi trình bày một vấn đề kinh điển khác, được sử dụng trong mật mã, xác định
cách tốt nhất để thu hoạch nguyên liệu thô và nhiều ứng dụng khác.

Lần này chúng tôi chỉ cung cấp một phương pháp khả thi để chuyển vấn đề thành một bài toán phù hợp chức năng và nhiễm sắc thể, cũng như một cách mới để phân bổ xác suất cho quá trình chọn lọc
nhà điều hành. Chúng tôi mời người đọc điền phần còn lại và tạo ra một thuật toán hoàn chỉnh để giải quyết nó. Ví dụ này được phỏng theo sách bài tập của Steeb [10].

Giả sử bạn là một người đi bộ đường dài đang lên kế hoạch cho một chuyến đi du lịch bụi và bạn có thể thoải mái mang theo bất cứ thứ gì.
hơn 20kg. Sau khi đã chọn được tất cả những đồ dùng bạn muốn mang theo, bạn
thấy rằng cộng lại chúng nặng hơn 20kg. Sau đó, bạn gán giá trị cho mỗi mục trong số 12 mục, được hiển thị trong Bảng 6, để giúp bạn lựa chọn nên thực hiện điều gì. Bạn nên mang theo những món đồ nào bạn có thể tối đa hóa giá trị của những gì bạn mang theo mà không vượt quá 20kg?

Một cách để dịch các phần tử thành nhiễm sắc thể là tạo các chuỗi bit có độ dài 12,
trong đó mỗi vị trí bit đại diện cho một mục cụ thể. Hãy để 1 chỉ ra rằng bạn bao gồm
mục cụ thể và 0 cho biết bạn loại trừ nó.

Bảng 6: Giá trị và trọng lượng của đồ dùng cắm trại

Mục	Trọng lượng giá trị
Thuốc đuổi côn	12 2
trùng bếp	4
trại căng tin	5 7
(đầy	10 5
đủ) quần áo	11 50 3
thực phẩm	15 3
khô bộ sơ	2
cứu	6 2
đèn pin	2
mới lạ áo mưa túi ngủ	4 3
lều	5 11
thiết bị lọc nước	25 20 30 1

Dựa trên điều này, hàm thích nghi của bạn phải biết trọng số và giá trị cố định liên quan với mỗi vị trí bit, vì nhiễm sắc thể không mang thông tin này. Thể lực của bạn Hàm cũng phải có khả năng cộng đồng thời hai tổng: tổng các giá trị của các mục để xác định mức độ phù hợp và tổng trọng lượng của các mặt hàng để đáp ứng giới hạn trọng lượng. [10]. Trong khi hàm thích nghi đang tổng hợp các giá trị và trọng số, có một cách để đảm bảo rằng nhiễm sắc thể nằm trong giới hạn trọng lượng là dừng ngay việc tính tổng và gán một độ thích nghi là 0 hoặc 1 nếu trọng lượng của nhiễm sắc thể vượt quá 20kg.

Cho đến nay chúng ta đã thảo luận hai phương pháp để tạo ra phân bố xác suất của toán tử lựa chọn: sử dụng các phân số của tổng mức độ phù hợp (trong Phần 1 và Ví dụ 2.1) và trọng số xếp hạng (trong Ví dụ 2.3 và Mục 4.1.2). Một chiến lược khác sử dụng softmax để tạo một phân phối tương tự như phương pháp đầu tiên của chúng ta, ngoại trừ việc nó có thể được sử dụng khi một số (hoặc tất cả) giá trị thích hợp là âm. Thay vì phân chia sức khỏe của nhiễm sắc thể giá trị bằng tổng các giá trị thích hợp của toàn bộ quần thể, hàm mũ của nhiễm sắc thể giá trị thích hợp được chia cho tổng các số mũ của các giá trị thích hợp của tổng thể. ĐẾN minh họa, xác suất một nhiễm sắc thể C53 được chọn để sinh sản sẽ là

$$P(C53) = \frac{e^{f(C53)}}{\sum_{tôi=1}^{Npop} e^{f(Ci)}} .$$

5 Thảo luận & Kết luận

Ngày nay, các thuật toán di truyền được các công ty lớn sử dụng để tối ưu hóa lịch trình và thiết kế các sản phẩm từ máy bay lớn đến chip máy tính nhỏ cho đến thuốc [6]. Phòng thí nghiệm quốc gia Los Alamos đã sử dụng chúng để phân tích dữ liệu từ vệ tinh [6]. Chúng được sử dụng để tạo ra những hiệu ứng đặc biệt chân thực cho những bộ phim như Troy và The Lord of the Nhẫn [6]. Ngoài ra còn có nhiều ứng dụng của thuật toán di truyền trong lĩnh vực tài chính, chẳng hạn như xác định gian lận và dự đoán biến động thị trường. Chúng thậm chí còn được sử dụng trong sự phát triển của các loại hình nghệ thuật và âm nhạc mới [3].

Tính toán song song đã làm cho các thuật toán di truyền trở nên hấp dẫn hơn bởi vì nó có thể giảm bớt nhược điểm chính của họ: thiếu tốc độ. Điều này cho phép lặp lại nhiều hơn và hơn thế nữa tổng số lần chạy, điều này làm tăng cả cơ hội tìm ra giải pháp tốt hơn và sự chắc chắn rằng các giải pháp tìm được là tối ưu.

Các nhà khoa học máy tính đang nghiên cứu tìm ra những cách mới để kết hợp các thuật toán di truyền với các thuật toán tối ưu hóa khác cũng như các nhánh tính toán tiến hóa khác, chẳng hạn như mạng lưới thần kinh [3]. Một ví dụ cơ bản kết hợp thuật toán di truyền với thuật toán dựa trên phép tính phương pháp leo đồi. Mỗi nhiễm sắc thể thích hợp nhất mà thuật toán di truyền tìm thấy là sau đó được sử dụng làm điểm khởi đầu cho thuật toán leo đồi, tuân theo độ dốc của hàm thích nghi từ đó đến mức tối ưu cục bộ [10]. Một lĩnh vực nổi bật và tương đối mới là lập trình di truyền, được phát minh vào năm 1989 bởi John Koza [4]. Lĩnh vực này mở rộng trên khuôn khổ của các thuật toán di truyền để cho phép các giải pháp ứng cử viên là phi tuyến tính, có các mức độ khác nhau kích thước và thậm chí cả các chương trình máy tính có thể thực thi được [4].

Đáng chú ý, điều vẫn còn cần được khám phá là lý thuyết đằng sau cách các thuật toán di truyền công việc. Trong cuốn sách nổi tiếng Thích ứng trong các hệ thống tự nhiên và nhân tạo, John Holland trình bày “Giả thuyết khối xây dựng”, trong đó nêu rằng các thuật toán di truyền hoạt động bằng cách “khám phá, nhấn mạnh và kết hợp lại các khối xây dựng tốt” [8]. Một khối xây dựng được định nghĩa là một chuỗi các giá trị bit (vì giả thuyết này được đề xuất khi nhiễm sắc thể luôn được định nghĩa là các chuỗi bit) ngắn hơn độ dài của nhiễm sắc thể,

mang lại mức độ phù hợp cao hơ n cho tất cả các nhiễm sắc thể mà nó hiện diện [8]. Đây vẫn là lý thuyết nổi bật cho đến những năm 1990, khi các bài báo đư ợc xuất bản bắt đầu xuất hiện đặt câu hỏi tính hợp lệ của nó. Điều kỳ lạ là trong khi các tác giả hiện nay đã giải thích cặn kẽ rằng Giả thuyết khối xây dựng giả định quá nhiều và sai, không ai đư a ra đư ợc phư ơ ng án khả thi giả thuyết thay thế. Có lẽ là do con ngư ời đã thấy các thuật toán di truyền hoạt động, và thế là đủ lời giải thích cho họ. Gần cuối cuốn sách của mình, Goldberg tuyên bố, “Thiên nhiên quan tâm đến những gì hiệu quả. Thiên nhiên truyền bá những gì tồn tại đư ợc. Cô ấy có rất ít thời gian để suy ngẫm một cách uyên bác, và chúng tôi đã cùng cô ấy theo đuổi mục tiêu cải thiện” [1].

Ngư ời giới thiệu

[1] Goldberg, DE (1989). Thuật toán di truyền trong tìm kiếm, tối ư u hóa và máy Học hỏi. Đọc: Addison-Wesley.

[2] Haupt, RL, & Haupt, SE (1998). Thuật toán di truyền thực tế. New York: Wiley-Liên khoa học.

[3] Haupt, RL, & Haupt, SE (2004). Thuật toán di truyền thực tế (tái bản lần thứ 2). Đi lang thang-ken: Wiley.

[4] Kinnear, KE (1994). Một góc nhìn về công việc trong cuốn sách này. Ở KE Kinnear (Ed.), Những tiến bộ trong lập trình di truyền (trang 3-17). Cambridge: Nhà xuất bản MIT.

[5] Koza, JR (1994). Giới thiệu về lập trình di truyền. Trong KE Kinnear (Ed.), Những tiến bộ trong lập trình di truyền (trang 21-41). Cambridge: Nhà xuất bản MIT.

[6] Mitchell, M. (2009). Độ phức tạp: Chuyến tham quan có hư ớng dẫn. New York: Nhà xuất bản Đại học Oxford.

[7] Mitchell, M. (1996). Giới thiệu về thuật toán di truyền. Cambridge: Nhà xuất bản MIT.

[8] Mitchell, M. (1995). Thuật toán di truyền: Tổng quan. Độ phức tạp, 1(1), 31-39.

[9] Simon, D. (2013). Thuật toán tối ưu hóa tiến hóa: Lấy cảm hứng từ sinh học và
Phương pháp tiếp cận dựa trên dân số đối với trí tuệ máy tính. Hoboken: Wiley.

[10] Steeb, W. (2001). Sách bài tập phi tuyến: Hỗn loạn, Phân số, Tự động hóa di động,
Mạng nơ-ron, Thuật toán di truyền, Logic mờ: với C++, Java, SymbolicC++
và Giảm bớt các chương trình. Singapore: Khoa học thế giới.

Phụ lục

Mã MATLAB cho ví dụ 2.2

Bài toán %: Xác định tập hợp dưới dạng chuỗi nhị phân có độ dài 20, với hàm thích nghi %: Tổng của 1. Sử dụng dân số 100 với xác suất % nhất định được xác định bên dưới để xem phải mất bao lâu để đạt được chuỗi % của tất cả 1. % % Haupt và Haupt, 2003 % Biên tập và sửa đổi bởi Hundley và Carr,

2014

```
%% I. Thiết lập các tham số GA
ff=inline('sum(x,2)'); % chức năng tập thể dục

đa=9999999;            % số lần lặp tối đa (đối với tiêu chí dừng) maxit=200; chi phí tối
% chi phí tối đa cho phép (đối với tiêu chí dừng) popsize=100; % quy mô quần thể
được đặt thay đổi = 0,001; % đặt tỷ lệ đột
biến

nbits=20; % số bit trong mỗi tham số npar=1; % số thông
số trong mỗi nhiễm sắc thể
Nt=nbits*npar; % tổng số bit trong nhiễm sắc thể

%% II. Tạo quần thể ban đầu iga=0; Bộ đếm thế
hệ % được khởi tạo pop=round(Rand(popsize,Nt)); %
quần thể ngẫu nhiên của số 1 và số 0 (ma trận 100 x 20)

% Khởi tạo chi phí và các mục khác để thiết lập vòng lặp chính
cost=feval(ff,pop); % tính toán chi phí dân số bằng cách sử dụng ff
 [cost,ind]=sort(cost,'descend'); % phần tử tối đa trong mục nhập đầu tiên
pop=pop(ind,:); % sắp xếp dân số có giá tối đa trước
```

```
maxc(1)=max(chi phí); % minc chứa minc tổng thể trung
bình(1)=mean(chi phí); % giá trị trung bình chứa giá trị trung bình của dân số

probs=chi phí/tổng(chi phí); % chuẩn hóa đơ n giản cho xác suất

%%III. VÒNG LẶP CHÍ NH

trong khi iga<maxit

     iga=iga+1; Bộ đếm tạo % tăng

% Chọn bạn tình
     M=trần(popsize/2); % số lần giao phối
     ma=RandChooseN(probs,M); % mate #1
     pa=RandChooseN(probs,M); % mate #2 % ma và
     pa chứa các chỉ số của nhiễm sắc thể sẽ giao phối

     xp=ceil(Rand(1,M)*(Nt-1)); % điểm giao nhau
     pop(1:2:popsize,:)=[pop(ma,1:xp(1)) pop(pa,xp(1)+1:Nt)]; % con đầu tiên
     pop(2:2:popsize,:)=[pop(pa,1:xp(1)) pop(ma,xp(1)+1:Nt)]; % con thứ hai

     % Thay đổi quần thể
     nmut=ceil((popsize-1)*Nt*mutrate); % tổng số đột biến
     mrow=ceil(Rand(1,nmut)*(popsize-1))+1; % hàng để biến đổi
     mcol=ceil(Rand(1,nmut)*Nt); cột % cần thay đổi
     với ii=1:nmut
          pop(mrow(ii),mcol(ii))=abs(pop(mrow(ii),mcol(ii))-1); % chuyển đổi bit cuối % ii


%% IV. Tổng thể đư ợc đánh giá lại theo chi phí
     cost=feval(ff,pop); % tính toán chi phí dân số bằng cách sử dụng hàm thích nghi
      [cost,ind]=sort(cost,'descend'); % phần tử tối đa trong mục nhập đầu tiên
     pop=pop(ind,:); % sắp xếp quần thể có chi phí tối đa đầu tiên
     maxc(iga+1)=max(cost);
     trung bình(iga+1)=trung
     bình(chi phí); probs=chi phí/tổng(chi phí);

%% V. Tiêu chí dừng nếu
     iga>maxit || chi phí(1)>chi phí tối đa

     kết thúc
%[iga cost(1)] Bỏ ghi chú này nếu bạn muốn theo dõi tiến trình cho %probs end %iga lớn
hơ n
```

```
%%VI. Hiển thị đầu ra như  trong Hình day=clock;

disp(datestr(datenum(day(1),day(2),day(3),day(4),day(5),day(6)),0)) định dạng ngắn g
disp(['popsize='
num2str(popsize) 'mutrate=' num2str(mutrate) '# par=' num2str(npar)]); disp(['#thế hệ = ' num2str(iga) ' giá tốt nhất
= fprintf('giải pháp tốt nhất\n%s\n',mat2str(int8(pop(1,:))));       ' num2str(chi phí(1))]);
figure(1) iters=0 :length(maxc)-1; cốt truyện(1:(iga+1),maxc,1:

(iga+1),meanc);
xlabel('Generation');ylabel('Cost');
```

B Mã MATLAB cho ví dụ 2.3

B.1 Định nghĩa hàm thích hợp

```
hàm elev=hàm kiểm tra(loc)
% Hàm chi phí cho ví dụ kinh độ-vĩ độ %

% Carr 2014

 [m,n]=kích
thư ớc(loc);
      for

      i=1:mx=loc(i,1); nếu x>10 x=10;
      nếu không x<0
            x=0;
      kết thúc
      y=loc(i,2);
      nếu y>10
            y=10;
      nếu không y<0
            y=0;
      kết thúc

      elev(i)=x*sin(4*x)+1.1*y*sin(2*y);
kết thúc
```

B.2 Thuật toán chính

```
%% Thuật toán di truyền liên tục %
```

```
% giảm thiểu hàm mục tiêu được chỉ định trong ff %

% Trước khi bắt đầu thiết lập đầy đủ các thông số ở phần I, II, III
% Haupt & Haupt 2003
% Biên tập bởi Hundley và Carr, 2014

%% Tôi thiết lập GA
ff='testfunction'; % hàm mục tiêu npar=2; % số
biến tối ưu hóa varhi=10; varlo=0; % giới hạn biến
đổi

%% II Tiêu chí dừng
maxit=200; % số lần lặp tối đa
mincost=-9999999; % giá trị nhỏ nhất

%% III Tham số GA
popsize=12; % quy mô quần thể được đặt
thay đổi=.2; % đặt lựa chọn tỷ lệ đột
biến = 0,5; % phần dân số được giữ Nt=npar; % tham
số liên tục GA Nt=#variables keep=floor(selection*popsize);
% #thành viên dân số sống sót nmut=ceil((popsize-1)*Nt*mutrate); % tổng số đột
biến M=ceil((popsize-keep)/2); % số lần giao phối



%% Tạo quần thể ban đầu iga=0; % bộ
đếm thế hệ được khởi tạo par=(varhi-
varlo)*rand(popsize,npar)+varlo; % ngẫu nhiên

Coords{1}=par;

cost=feval(ff,par); % tính toán chi phí dân số bằng cách sử dụng ff
 [cost,ind]=sort(cost); % chi phí tối thiểu trong phần
tử 1 par=par(ind,:); % sắp xếp liên tục
minc(1)=min(cost); % minc chứa minc(1)=mean(cost);
% giá trị trung bình chứa giá trị trung bình của dân số

%% Lặp lại qua các thế hệ (Vòng lặp chính) while
iga<maxit
     iga=iga+1; Bộ đếm tạo % tăng

     %_____________________________________________________________
     % Ghép đôi và giao phối
     M=ceil((popsize-keep)/2); % số lần giao phối
     prob=flipud([1:keep]'/sum([1:keep])); % trọng lượng tỷ lệ nhiễm sắc
     thể=[0 cumsum(prob(1:keep))']; hàm phân bố xác suất %
```

```
pick1=rand(1,M); % mate #1 (vectơ có độ dài M với các số ngẫu nhiên trong khoảng từ 0 đến 1) pick2=rand(1,M);
% bạn #2

% ma và pa chứa chỉ số của nhiễm sắc thể sẽ giao phối
% Chọn số nguyên k với xác suất p(k) % ic=1;


trong khi ic<=M
      cho id=2:keep+1 nếu
            pick1(ic)<=odds(id) && pick1(ic)>odds(id-1) ma(ic)=id-1;

            kết thúc
            nếu pick2(ic)<=odds(id) && pick2(ic)>odds(id-1)
                  pa(ic)=id-1;
            kết thúc
      kết thúc
      ic=ic+1;
kết thúc


%_____________________________________________________________
% Thực hiện giao phối bằng cách sử dụng chéo điểm đơ n

ix=1:2:giữ; % chỉ số của bạn đời #1
xp=ceil(rand(1,M)*Nt); % điểm giao nhau r=Rand(1,M);
thông số trộn %

với ic=1:M

      xy=par(ma(ic),xp(ic))-par(pa(ic),xp(ic)); % ma và bố

      par(keep+ix(ic),:)=par(ma(ic),:); % Con thứ nhất par(keep+ix(ic)
      +1,:)=par(pa(ic),:); % con thứ 2

      par(keep+ix(ic),xp(ic))=par(ma(ic),xp(ic))-r(ic).*xy; % mệnh giá thứ nhất(keep+ix(ic)
      +1,xp(ic))=par(pa(ic),xp(ic))+r(ic).*xy; % lần 2

      if xp(ic)<npar % chéo khi biến cuối cùng không đư ợc chọn
            par(keep+ix(ic),:)=[par(keep+ix(ic),1:xp(ic)) par(keep+ix(ic )
            +1,xp(ic)+1:npar)]; par(keep+ix(ic)
            +1,:)=[par(keep+ix(ic)+1,1:xp(ic)) par(keep+ix(ic),xp(ic)+1:npar )];
            kết thúc % nếu

kết thúc
```

```
      %_____________________________________________________________
      % Thay đổi quần thể
      mrow=sort(ceil(Rand(1,nmut)*(popsize-1))+1);
      mcol=ceil(rand(1,nmut)*Nt); với
      ii=1:nmut
           par(mrow(ii),mcol(ii))=(varhi-varlo)*rand+varlo; % kết thúc
      đột biến %
      ii
      %_____________________________________________________
      % Con cái mới và nhiễm sắc thể đột biến đư ợc đánh giá %


      cost=feval(ff,par);
      %_____________________________________________________________
      % Sắp xếp chi phí và các tham số liên quan
       [cost,ind]=sort(cost);
      par=par(ind,:);
      Tọa độ{iga+1}=par;

      %_____________________________________________________________
      % Thực hiện thống kê cho một lần chạy không tính trung
      bình minc(iga+1)=min(cost);
      trung bình(iga+1)=trung
      bình(chi phí); %_____________________________________________________________
      % Tiêu chí dừng nếu
      iga>maxit || chi phí(1)<chi phí tối
           thiểu
      kết thúc
       [chi phí iga(1)];
kết thúc %iga


%% Hiển thị ngày đầu
ra=đồng hồ;
disp(datestr(datenum(day(1),day(2),day(3),day(4),day(5),day(6)),0)) disp(['chức
năng đư ợc tối ư u hóa là ' ff] ) định dạng
ngắn g
disp(['popsize=' num2str(popsize) ' mutrate=' num2str(mutrate) ' # par=' num2str(npar)]) disp(['#thế hệ='
num2str(iga) ' best cost= ' num2str(cost(1))]) disp('giải pháp tốt nhất')
disp(num2str(par(1,:)))
disp('thuật toán di truyền
liên tục')

hình(1)
iters=0:length(minc)-1;
```

```
cốt truyện(iters,minc,iters,meanc,'-');
xlabel('thế hệ');ylabel('chi phí');
```

Mã MATLAB cho bài toán người bán hàng du lịch Thuật toán di truyền, Phần 4.1.2

C.1 Định nghĩa hàm thích hợp

```
hàm dist=tspfun(pop)
Hàm % chi phí cho bài toán nhân viên bán hàng du lịch.
% Sử dụng các biến toàn cục "x" và "y" % %

Haupt và Haupt, 2003 % Chỉnh
sửa và sửa đổi bởi Hundley và Carr, 2014

xy toàn cầu

 [Npop,Ncity]=size(pop);
tour=[pop pop(:,1)];

%khoảng cách giữa các thành
phố cho ic=1:Ncity
     cho id=1:Ncity
          dcity(ic,id)=sqrt((x(ic)-x(id))^2+(y(ic)-y(id)) ^2); cuối % id
     cuối % ic



% chi phí của mỗi nhiễm sắc thể
cho ic=1:Npop
     dist(ic,1)=0;
     for id=1:Ncity
          dist(ic,1)=dist(ic)+dcity(tour(ic,id),tour(ic,id+1)); cuối % id
     cuối % ic
```

C.2 Thuật toán chính

```
%% Thuật toán di truyền cho các bài toán hoán vị % giảm
thiểu hàm mục tiêu được chỉ định trong ff %

% Haupt và Haupt, 2003
% Biên tập và sửa đổi bởi Hundley và Carr, 2014
```

thông thoáng

```
iga xy toàn cầu

%% Thiết lập GA
ff='tspfun'; % tên tệp hàm mục tiêu npar=20; % # biến
tối ư u hóa % # cột trong ma trận tổng thể
Nt=npar;



x=rand(1,npar);
y=rand(1,npar); % thành phố ở (xcity,ycity)

% Bỏ ghi chú dòng tiếp theo để sử dụng cùng một nhóm thành phố trong nhiều lần chạy %
tải thành phố0

% Tiêu chí dừng
maxit=10000; % số lần lặp tối đa

% tham số GA
popsize=20; % quy mô quần thể đư ợc đặt
thay đổi=.05; % đặt lựa chọn tỷ lệ đột
biến = 0,5; % phần dân số đư ợc giữ lại

keep=sàn(lựa chọn*popsize); % #dân số sống sót

     M=ceil((popsize-keep)/2); % số lần giao phối = 1; for
     ii=2:giữ
     tỷ lệ cư ợc=[tỷ
          lệ cư ợc ii*ones(1,ii)];
     kết thúc
     Gật đầu=độ dài(tỷ lệ
     cư ợc); tỷ lệ cư ợc=giữ tỷ lệ cư ợc+1; % tỷ lệ cư ợc xác định xác suất làm cha mẹ

% Tạo quần thể ban đầu iga=0; Bộ đếm
thế hệ % đư ợc khởi tạo cho iz=1:popsize

pop(iz,:)=randperm(npar); % dân số ngẫu nhiên
kết thúc

chi phí=feval(ff,pop);       % tính chi phí dân số bằng cách sử dụng ff

 [chi phí,ind]=sắp xếp(chi phí); % chi phí tối thiểu trong
phần tử 1 pop=pop(ind,:); % sắp xếp quần thể có chi phí thấp nhất đầu tiên
minc(1)=min(cost); % minc chứa minc tổng thể trung bình(1)=mean(chi
phí); % giá trị trung bình chứa giá trị trung bình của dân số
```

```
%% Lặp lại qua các thế hệ (VÒNG CHÍ NH) while
iga<maxit
iga=iga+1; Bộ đếm tạo % tăng

% Ghép đôi và giao
phối pick1=ceil(Gật đầu*rand(1,M)); % mate #1
pick2=ceil(Gật đầu*rand(1,M)); % mate #2 % ma
và pa chứa các chỉ số của cha mẹ ma=odds(pick1); pa=tỷ
lệ cư ợc(pick2);


% Thực hiện giao
phối cho
ic=1:M mate1=pop(ma(ic),:);
mate2=pop(pa(ic),:);
indx=2*(ic-1)+1; % số lẻ bắt đầu từ 1
xp=ceil(rand*npar); % giá trị ngẫu nhiên trong khoảng từ 1
đến N

temp=mate1; x0=xp; trong khi
mate1(xp)~=temp(x0)
mate1(xp)=mate2(xp);
mate2(xp)=temp(xp);

xs=find(temp==mate1(xp)); xp=xs; kết thúc
pop(keep+indx,:)=mate1;
pop(keep+indx+1,:)=mate2; kết
thúc


% Thay đổi quần thể
nmut=ceil(popsize*npar*mutrate);
với ic = 1:nmut
row1=ceil(Rand*(popsize-1))+1;
col1=ceil(rand*npar);
col2=ceil(rand*npar);
temp=pop(row1,col1);
pop(row1,col1)=pop(row1,col2);
pop(row1,col2)=temp;
im(ic)=row1;
Kết thúc
chi phí=feval(ff,pop);
%_____
% Sắp xếp chi phí và các tham số liên quan part=pop;
```

```
costt=chi phí;
 [chi phí,ind]=sắp xếp(chi
phí); pop=pop(ind,:);
%_____________________________________________________________
% Thực hiện thống
kê minc(iga)=min(cost);
Meanc(iga)=trung bình(chi
phí); kết thúc %iga

%_____________________________________________________________
% Hiển thị kết quả ngày=đồng
hồ;
disp(datestr(datenum(day(1),day(2),day(3),day(4),day(5),day(6)),0)) disp(['chức năng được
tối ưu hóa là ' ff] ) định dạng ngắn g
disp(['popsize='
num2str(popsize) 'mutrate=' num2str(mutrate) '# par=' num2str(npar)]) disp([' best cost=' num2str(cost(1))] )
disp(['giải pháp tốt nhất']); disp([num2str(pop(1,:))])


hình(2)
   iters=1:maxit; cốt
   truyện(iters,minc,iters,meanc,'--');
   xlabel('thế hệ'); ylabel('chi
   phí');

Hình 1); cốt
   truyện([x(pop(1,:)) x(pop(1,1))],[y(pop(1,:)) y(pop(1,1))],x,y,'o '); trục vuông
```