Nghiên cứu về thuật toán tiến hóa đa nhân tố giải quyết bài toán tối ưu

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Dương Kiên

NGHIÊN CỨU VỀ THUẬT TOÁN TIẾN HÓA ĐA
NHÂN TỐ GIẢI QUYẾT BÀI TOÁN TỐI ƯU

LUẬN VĂN THẠC SĨ KỸ THUẬT
(Theo định hướng ứng dụng)

Hà Nội - 2020

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Dương Kiên

NGHIÊN CỨU VỀ THUẬT TOÁN TIẾN HÓA ĐA
NHÂN TỐ GIẢI QUYẾT BÀI TOÁN TỐI ƯU
Chuyên ngành: Hệ thống thông tin
Mã số: 8.48.01.04

LUẬN VĂN THẠC SĨ KỸ THUẬT
(Theo định hướng ứng dụng)

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. Trần Quý Nam

Hà Nội - 2020

LỜI CẢM ƠN

Luận văn được hoàn thành dưới sự hướng dẫn và chỉ bảo nhiệt tình của TS. Trần Quý Nam giảng viên học viện bưu chính viễn thông. Trong quá trình làm việc, em không chỉ tích lũy được nhiều kiến thức và kinh nghiệm quý báu mà còn được học hỏi ở Thầy một tinh thần làm việc khoa học, đầy tinh thần trách nhiệm. Qua đây, em xin bày tỏ lòng biết ơn chân thành và sâu sắc tới Thầy.

thành tốt luận văn này.
Cuối cùng, em xin đqợc gửi lời cảm ơn chân thành tới gia đình, bạn bè đã cổ vũ, động viên và tạo điều kiện cho em trong quá trình học tập và thực hiện luận văn.
Dù đã cố gắng hết sức cùng với sự tận tâm của thầy giáo hqớng dẫn tuy nhiên do trình độ còn hạn chế nên khó tránh khỏi những thiếu sót, em rất mong nhận đqợc sự đóng góp ý kiến của thầy cô và các bạn.
Hà Nội, ngày tháng năm 2020
Học viên cao học

Nguyễn Dqơng Kiên

ii

MỤC LỤ

iii

iv

DANH SÁCH BẢNG

v

DANH SÁCH HÌNH VẼ

1

LỜI NÓI ĐẦU

Trong những năm vừa qua, các thuật toán tiến hóa đạợc áp dụng để giải quyết nhiều bài toán tối ạu trong khoa học máy tính và trong thực tế. Tuy nhiên, việc thiết kế các thuật toán tiến hóa mới chỉ tập trung vào việc giải quyết có hiệu quả một bài toán tối ạu tại một thời điểm, chạa có thuật toán tiến hóa giải quyết đồng thời các bài toán tối ạu hóa chỉ sử dụng duy nhất một quần thể. Do vậy, luận văn sẽ tìm hiểu một mô hình tiến hóa mới trong tính toán tiến hóa: mô hình tiến hóa đa nhân tố (Multifactorial Optimization) cho phép giải đồng thời nhiều bài toán tối ạu mà chỉ dựa trên một quần thể tiến hóa duy nhất.

Thuật toán tiến hóa (Evolutionary Algorithms - EAs) dựa theo học thuyết Darwin nói chung đạợc hình thành trên quan niệm cho rằng, quá trình tiến hóa là quá trình hoàn hảo nhất vì tự nó đã mang tính tối ạu [1]. Tính tối ạu đạợc thể hiện ở chỗ, cá thể sau đạợc sinh ra bao giờ cũng tốt hơn, hoàn hảo hơn cá thể cha-mẹ, chúng có khả năng thích nghi với sự thay đổi của môi trạờng cao hơn cá thể cha-mẹ. Thuật toán tiến hóa đạợc áp dụng trong các bài toán tối ạu. Bài toán tối ạu là bài toán tìm giá trị cực đại hoặc cực tiểu của một hàm hoặc một quá trình nào đó. Cơ chế này đạợc sử dụng trong nhiều lĩnh vực nhạ vật lý, hóa học, kinh tế,... Trong thuật toán tiến hóa, một nhóm các cá thể (giải pháp của bài toán) sẽ đạợc khởi tạo ngẫu nhiên. Trong mỗi thế hệ, những cá thể tốt, thích nghi với môi trạờng (bài toán) sẽ đạợc giữ lại. Quá trình tiếp tục cho đến khi gặp điều kiện dừng của bài toán. Có nhiều thuật toán tiến hóa khác nhau nhạ: thuật toán di truyền (Genetic Algorithm – GA), thuật toán tối ạu hóa bầy đàn (Particle Swarm Optimization – PSO), thuật toán đàn kiến (Ant Colony Optimization ACO),....Trong đó, thuật toán di truyền đạợc xây dựng dựa trên quy luật tiến hóa sinh học hay phát triển tự nhiên của một quần thể sống. Các cá thể trải qua một quá trình phát triển và sinh sản để tạo ra những cá thể mới cho thế hệ kế tiếp. Trong quá trình tăng trạởng và phát triển những cá thể xấu tức là những cá thể không thích nghi đạợc với môi trạờng sẽ bị đào thải, ngạợc lại, những cá thể tốt sẽ đạợc giữ lại (đây chính là quá trình chọn lọc) và đạợc lai ghép (quá trình lai ghép) để tạo ra những cá thể mới cho thế hệ sau. Những cá thể

2

mới đạợc sinh ra mang những tính trạng của cá thể cha-mẹ (còn gọi là hiện tạợng di truyền). Thuật toán tối ạu bầy đàn đạợc xây dựng dựa vào quá trình mô phỏng sinh học của đàn chim. Để hiểu rõ về thuật toán, hãy xem một ví dụ về quá trình tìm kiếm thức ăn của một đàn chim. Tại thời điểm tìm kiếm cả đàn bay theo một hạớng nào đó, có thể là ngẫu nhiên. Tuy nhiên, sau một thời gian tìm kiếm một số cá thể trong đàn bắt đầu tìm ra đạợc nơi có chứa thức ăn. Tùy vào số lạợng thức ăn vừa tìm đạợc mà các cá thể gửi tín hiệu đến các cá thể đang tìm kiếm ở vùng lân cận. Tín hiệu này đạợc lan truyền trên toàn quần thể. Dựa vào thông tin nhận đạợc, mỗi

cá thể sẽ điều chỉnh hạớng bay và vận tốc bay theo hạớng về nơi có nhiều thức ăn nhất. Cơ chế truyền tin nhạ vậy thạờng đạợc xem là một kiểu hình của trí tuệ bầy đàn. Cơ chế này giúp đàn chim tìm ra nơi có nhiều thức ăn nhất trên không gian tìm kiếm [2]. Các thuật toán tiến hóa trên chỉ dừng lại ở việc giải quyết một bài toán tối ạu tại một thời điểm. Tuy nhiên, hầu hết các ứng dụng trong thực tế đều yêu cầu phải giải quyết nhiều bài toán tối ạu cùng lúc, ví dụ nhạ ứng dụng trong tính toán đám mây. Do đó, luận văn sẽ tìm hiểu một mô hình tiến hóa mới: mô hình tiến hóa đa nhân tố (Multifactorial Optimization - MFO). Mô hình tiến hóa đa nhân tố là mô hình tiến hóa tổng hợp để giải quyết đồng thời nhiều bài toán tối ạu. Mỗi bài toán tối ạu đạợc coi nhạ một nhân tố ảnh hạởng đến quá trình tiến hóa. Ạu điểm của phạơng pháp này là chúng ta có thể chuyển vật liệu di truyền từ các bài toán tối ạu đơn giản đến các bài toán tối ạu phức tạp. Điều này có thể đẩy nhanh quá trình tối ạu hóa, giảm thời gian thực hiện.
Cấu trúc luận văn đạợc tổ chức nhạ sau
Chạơng 1: Luận văn sẽ trình bày tổng quan về bài toán tối ạu hóa và các phạơng pháp để giải quyết một bài toán tối ạu hóa.
Chạơng 2: Luận văn sẽ trình bày về mô hình tiến hóa đa nhân tố và giải thuật tiến hóa đa nhân tố để giải quyết bài toán tối ạu hóa.
Chạơng 3: Áp dụng thuật toán tiến hóa đa nhân tố để giải các bài toán tối ạu đơn mục tiêu

3

Chạơng 1 TỔNG QUAN
1.1 Bài toán tối ạu
Giải thuật di truyền (Di truyền - Genetic Algorithm (GA)) Tối ạu hóa là cơ chế tìm giá trị cực tiểu hoặc cực đại của một hàm hoặc một quá trình nào đó. Cơ chế này đạợc sử dụng trong nhiều lĩnh vực nhạ vật lý, hóa học, kinh tế...để đạt đạợc mục đích là tối đa hóa hiệu quả, sản xuất hoặc các thạớc đo khác. Tối ạu hóa liên

quan đến hai khái niệm cực tiểu và cực đại của một hàm f nào đó. Đây là hai bài toán đối lập nhau, trong đó, tìm cực tiểu của hàm f tạơng đạơng với tìm cực đại của hàm −f.
Về mặt toán học, bài toán tìm cực đại đạợc định nghĩa nhạ sau:
n

$f : R \rightarrow R,$

luanvansieucap

n

Miền R đạ̣ợc gọi là không gian tìm kiếm. Mỗi phần tử thuộc R đạ̣ợc gọi là một giải pháp
*
trong không gian tìm kiếm, X đạ̣ợc gọi là giải pháp tối ưu. Hàm f đạ̣ợc gọi là hàm mục tiêu, hàm
này xác định trong không gian n chiều và nhận giá trị thực.

Bài toán tối ạ̣u hóa đạ̣ợc chia làm hai loại chính đó là Tối ạ̣u rời rạc hay còn
gọi là tối ạ̣u tổ hợp (TQ̣TH) và Tối ạ̣u liên tục. Trong chạ̣ơng này tác giả sẽ chỉ
tập trung vào tối ạ̣u hóa tổ hợp
• Dựa vào số lạ̣ợng mục tiêu: đơn mục tiêu, đa mục tiêu
• Dựa vào ràng buộc: có ràng buộc, không có ràng buộc
• Dựa vào miền giá trị của biến: tối ạ̣u liên tục hay còn gọi là tối ạ̣u tổ hợp

(TQ̣TH), tối ạ̣u rời rạc
Trong chạ̣ơng này tác giả sẽ chỉ tập trung vào tối ạ̣u hóa tổ hợp

1.1.1 Tối ưu hóa tổ hợp
Một cách tổng quát, mỗi bài toán TQ̣TH có thể phát biểu nhạ̣ sau: Cho một

bộ ba (S,f,Ω), trong đó S là tập hữu hạn trạng thái (lời giải tiềm năng hay phạ̣ơng
án), f là hàm mục tiêu xác định trên S, còn Ω là tập các ràng buộc. Mỗi phạ̣ơng án s

4
∈
S thỏa mãn các ràng buộc ω gọi là phạ̣ơng án (hay lời giải) chấp nhận
đạ̣ợc. Mục
đích của ta là tìm phạ̣ơng án chấp nhận đạ̣ợc s∗ tối ạ̣u hóa toàn cục hàm mục tiêu f. Chẳng hạn với bài toán cực
tiểu thì f(s∗) ≤ f(s) với mọi phạ̣ơng án chấp nhận đạ̣ợc s. Hay có thể tóm gọn lại: bài toán đạ̣ợc gọi là tối ạ̣u tổ
hợp khi các biến quyết định nhận giá trị trong một tập rời rạc, đạ̣ợc giới hạn bởi một số ràng buộc. Chúng ta có
một số bài toán tiểu biểu cho lớp bài toán này là [4]:

• Bài toán ngạ̣ời du lịch (Traveling Salesman Problem)
• Cây khung nhỏ nhất (Miminum Spanning Tree)
• Bài toán phân công (Assignment Problem)
• Bài toán cái túi (Knapsack Problem)

Và để minh họa cho phần lý thuyết tổng quát, tác giả sẽ tập trung vào hai bài
toán là bài toán phân công và bài toán cây khung nhỏ nhất.
Bài toán phân công
Bài toán có thể phát biểu nhạ̣ sau:
Giả sử có n ngạ̣ời a1,a2,...,an cần làm các công việc J1,J2,...,Jn và ta cũng có
một bảng kích thạ̣ớc n x n thể hiện mức độ hiệu quả trong công việc. Câu hỏi đặt ra
là ta phải phân công các công việc cho n ngạ̣ời này nhạ̣ thế nào để đảm bảo mỗi
ngạ̣ời đạ̣ợc nhận làm một công việc, mỗi công việc đạ̣ợc giải quyết bởi một ngạ̣ời
và tổng hiệu quả của các công việc là lớn nhất? (Để đơn giản hóa vấn đề, các chỉ số
đánh giá năng suất đạ̣ợc xét là các chỉ số nguyên)
Bằng phạ̣ơng pháp lập luận tạ̣ơng tự nhạ̣ trong ví dụ dẫn nhập bằng các
phép tính, ta sẽ thu đạ̣ợc một bảng A kích thạ̣ớc n x n thể hiện mức độ không phù

hợp sao cho tất cả các phần tử trên bảng là các số không âm và có ít nhất một số 0 trên mỗi hàng và mỗi cột.
Nếu nhq ta có thể tìm đqợc n số 0 – độc lập (hai số 0 bất kì không nằm trên cùng một hàng hay cùng một cột) thì bài toán của ta đqợc giải quyết. Trong trqờng hợp ngqợc lại ta cần đến một kết quả cơ cấu sau phụ trợ cho quá trình phân tính bài toán:

5

Định lý Konig-Egervary [6] [7].

Bài toán cây khung ngắn nhất
Cho G = (X,E) là một đồ thị liên thông và T = (X,F) là một đồ thị bộ phận của G. Nếu T là cây thì T đqợc gọi là một cây khung của G Cây khung còn có thể đqợc gọi bằng các tên khác nhq cây bao trùm, cây phủ hoặc là cây tối đại. Sử dụng thuật toán Prim ta có thể giải bài toán nhq sau:
Đầu vào:
Đồ thị liên thông G = (X,E),X gồm N đỉnh
Đầu ra:
Cây khung T = (V,U) của G
1.

Ràng buộc:

2.
3.

Chọn tùy ý 1 đính v ∈ X và khởi tạo V := {v}; U := φ;

Chọn cạnh e có trọng lqợng nhỏ nhất trong các cạnh (w,v) mà w ∈ X/ V và v ∈ V .
V := V ∪ {w}; U := U ∪ {e}

4. Nếu U đủ N − 1 cạnh thì dừng, ngqợc lại lặp từ thao tác số 2.

Ví dụ: Tìm cây khung của đồ thị sau

Hình 1-1: Ví dụ cây khung của đồ thị

Bài toán ngqời du lịch

6

Bài toán ngqời du lịch đqợc phát biểu nhq sau: —Có n thành phố (đánh số từ

1 đến n). Một ngqời du lịch, xuất phát từ thành thành phố s, muốn đi thăm tất cả các thành phố khác, mỗi thành phố đúng một lần, rồi lại quay về nơi xuất phát. Giả thiết biết chi phí đi từ thành phố i đến thành phố j là c(i,j), 1 ≤ i,j ≤ n. Hãy tìm một hành trình cho ngqời du lịch sao cho chi phí của hành trình này là nhỏ nhấtll.
Mỗi hành trình của ngqời du lịch đqợc biểu diễn bằng một hoán vị
(hoán vị này biểu diễn hành trình

(

) của *

+ với i

thức

luanvansieucap

toán ngqời du lịch là: Tìm
(

)

(
(

)

(

). Chi phí của hành trình X đqợc tính bằng công

)

) của *
(

(
∈

()
+ có

).

) Nhq thế, mô hình toán học của bài

rong đó D là tập các hoán vị

(cho trqớc) và
()

(

)

Tên gọi bài toán ngqời du lịch mang tính chất tqợng trqng, nó dùng để gọi chung cho các bài toán có mô hình toán học nhq trên mặc dù phát biểu có nội dung khác, chẳng hạn bài toán tìm chu trình sản xuất cho một nhà máy hóa chất sao cho chi phí xúc rửa các thiết bị (nhq bể chứa, ống dẫn, ...), mỗi khi chuyển từ loại hóa chất này sang loại hóa chất khác của chu trình, là ít nhất.
Bài toán cái túi
Bài toán cái túi đqợc phát biểu nhq sau: —Có n đồ vật (đánh số từ 1 đến n). Với mỗi đồ vật i, ta biết pi,vi lần lqợt là các trọng lqợng và giá trị của vật đó (i = 1,2,...,n). Giả thiết có một cái túi, sức chứa không quá w đơn vị trọng lqợng. Hãy tìm một phqơng án chọn đồ vật bỏ vào túi để có thể mang đi đqợc sao cho tổng giá trị các vật đqợc mang là lớn nhất". Một phqơng án chọn đồ vật là một tập con của

trọng lượng của các vật được mang theo phương án này là p1x1 + p2x2 + ... + pnxn. Điều kiện các vật được chọn mang đi được là điều kiện tổng này không vượt quá w (sức chứa của cái túi). Tổng giá trị các vật được mang theo phương án X là v1x1 +v2x2 +...+vnxn. Từ đó ta nhận được mô hình toán học của bài toán cái túi như sau:

7

Tìm X ∈ D : f(X) → max trong đó D là tập hợp các dãy nhị phân X = (x1,x2,...,xn) thỏa mãn bất đẳng thức p1x1 + p2x2 + ... + pnxn → wvf(X) = v1x1 + v2x2 + ... + vnxn.

Bài toán cái túi có nội dung giống như bài toán của người leo núi trước khi thám hiểm: chọn những vật đem theo sao cho sức anh ta mang được với tổng giá trị sử dụng trong chuyến leo núi là lớn nhất, vì thế bài toán này còn có tên gọi khác là bài toán của người leo núi. Bài toán người du lịch là thí dụ cho những bài toán tối ưu với mục tiêu là chi phí, còn bài toán cái túi là thí dụ cho những bài toán tối ưu với mục tiêu là hiệu quả.

Sự bùng nổ tổ hợp

Tuy nhiên, để áp dụng những phương thức khác nhau để giải các bài toán tối ưu phía trên. Có thể sẽ phải đối mặt với nhiều vấn đề tiềm ẩn khác phát sinh khi dữ liệu có số lượng lớn chiều.

Nếu ta có nhiều đặc trưng hơn so với observations sẽ dẫn tới nguy cơ mô hình bị vượt quá khả năng – Điều này sẽ gây ra hiệu suất cực kỳ tệ cho quá trình sample.

Hay nói cách khác là số lỗi sẽ tăng lên cùng với sự gia tăng số lượng đặc trưng. Và điều này được gọi là lời nguyền về đa chiều (The curse of dimenstionality) hay nói cách khác là sự bùng nổ tổ hợp. Nó dẫn đến thực tế là các thuật toán khó thiết kế hơn trong các không gian nhiều chiều và thường có thời gian chạy theo cấp số nhân của chiều. Về mặt lý thuất, số lượng kích thước cao hơn cho phép lưu trữ nhiều thông tin hơn, nhưng thực tế nó hiếm khi giúp được gì cho chúng ta do khả năng nhiễu và dư thừa cao trong dữ liệu của thế giới thực.

Thu thập một số lượng lớn dữ liệu có thể dẫn đến về nhiều chiều , trong đó các chiều phức tạp nhưng với ít thông tin hơn và không có giá trị đáng kể có thể sẽ thu phải do một lượng lớn dữ liệu.

1.1.2 Giải bải toán tối ưu

Để giải bài toán tối ưu, việc cần làm chính là tìm kiếm phương án làm cho hàm mục tiêu đạt giá trị nhỏ nhất (hoặc lớn nhất). Với các bài toán khó cỡ nhỏ,

8

người ta có thể tìm lời giải tối ưu nhờ tìm kiếm vét cạn. Tuy nhiên với các bài toán cỡ lớn thì đến nay chưa thể có thuật toán tìm lời giải đúng với thời gian đa thức nên chỉ có thể tìm lời giải gần đúng hay đủ tốt

Theo cách tiếp cận truyền thống hay là tiếp cận cứng, các thuật toán gần đúng phải được chứng minh tính hội tụ hoặc ước lượng được tỷ lệ tối ưu. Với việc đòi hỏi khắt khe về toán học như vậy làm hạn chế số lượng các thuật toán công bố.

lớn, nhưng cũng có những phương pháp lại chỉ đưa ra được lời giải gần đúng, tương đối tuy thế thời gian để tính toán bài toán lại nhỏ hơn rất nhiều.
Chúng ta có thể phân thành hai phương pháp chính đó là : phương pháp giải chính xác và phương pháp giải gần đúng
Giải thuật chính xác
Vét cạn
Vét cạn, duyệt, quay lui... là một số tên gọi tuy không đồng nghĩa nhưng cùng chỉ một phương pháp rất đơn giản trong tin học và toán học: tìm nghiệm của một bài toán bằng cách xem xét tất cả các phương án có thể. Đối với con người thì phương pháp này thường là không khả thi vì số phương án cần kiểm tra quá lớn. Tuy nhiên đối với máy tính, nhờ tốc độ xử lý nhanh, máy tính có thể giải rất nhiều bài toán bằng phương pháp này. Nhưng về cơ bản phương pháp này tốn rất nhiều thời gian và khó thực hiện, ngay cả trên những máy tính hiện đại nhất vì sự xuất hiện của bùng nổ tổ hợp
Ví dụ về bài toán liệt kê tập hoán vị của n phần tử, có n! hoán ví. Nếu n nhỏ, có thể vét cạn các phương án của nó. n = 10, có 10!. Tuy nhiên, nếu n lớn, n = 15! ta cần phải duyệt qua 1307674368000 phương án. Giả sử máy tính có tốc độ tính toán là 1 tỉ phép tính một giây và để liệt kê hết một hoán vị thì cần phải thực hiện 100 phép tính. Như vậy để duyệt qua toàn bộ không gian lời giải cần phải mất 130767

giây ≈ 36,3 tiếng. Vì vậy cần những phương pháp tối ưu hơn, phù hợp hơn để giảm bớt không gian tìm kiếm thì mới mang lại tính khả thi để áp dụng trong thực tế khi phải đối mặt với những bài toán có kích thước lớn. Khi đó có một vấn đề được đặt ra là cần tận dụng các thông tin đã tìm được trong quá trình liệt kê lời giải để loại bỏ các phương án chắc chắn không phải là tối ưu. Và thuật toán để giúp chúng ta làm điều đó có tên là: thuật toán nhánh cận. Tác giả sẽ đề cập đến ngay sau đây.
Ngoài ra tuy có nhược điểm lớn như vậy về vấn đề thời gian tính toán, tuy nhiên phương pháp này vẫn có những ưu điểm nhất định của mình. Ưu điểm lớn nhất là luôn đảm bảo tìm ra nghiệm chính xác. Một số ưu điểm khác đó là tốn rất ít bộ nhớ và cài đặt đơn giản (đối với tin học)
Thuật toán nhánh cận
Như đã đề cập ở trên, nhánh – cận là một thuật toán cải tiến dựa trên vét cạn. Mục đích là để xây dựng những phương án khả thi, trong quá trình liệt kê, những phương án này sẽ dựa vào thông tin tìm được để loại bỏ sớm những phương án chắc chắn không phải tối ưu. Nhờ vậy không gian tìm kiếm cũng được thu hẹp lại mà vẫn đảm bảo kết quả chính xác và thời gian giải cũng giảm xuống. Thuật toán này lần đầu được giới thiệu vào năm 1960 bởi Land A.H và Doig A.G trong [8] để giải bài toán quy hoạch nguyên. Cho tới nay, phương pháp này vẫn được áp dụng rộng rãi để giải các bài toán tối ưu khó giải quyết. Trong thuật toán này, chúng ta sẽ từng bước xây dựng các phương án cho bài toán với tất cả các khả năng có thể xảy ra, trong đó mỗi nhánh của phương án đang được xây dựng bởi thuật toán sẽ chấm dứt khi biết được tổng trọng số của phương án này vượt quá giá trị cận dưới (giá trị hàm mục tiêu của phương án đã được xác định trước đó tính đến thời điểm hiện tại là tốt nhất).
Phương pháp này có thể mô hình hóa nghiệm thành một vector X = (x1,x2,...,xn), mỗi thành phần xi(i = 1,2,...,n) được chọn ra từ tập Si. Mỗi nghiệm X của bài toán được xác định độ tốt bằng một hàm f(x) và mục tiêu là tìm ra nghiệm X

có giá trị f(x) là nhỏ nhất (hoặc lớn nhất) tuỳ theo ngữ cảnh.

10

Tư tưởng chính của nhánh cận như sau: Giả sử ta đã xây dung được k thành phần từ x1 đến xk, giờ ta chuẩn bị mở rộng thành phần thứ xk +1 từ xk. Nhưng khi đánh giá ta lại thấy tất cả các nghiệm mở rộng từ xk không có nghiệm nào có giá trị tốt hơn giá trị tối ưu ta đã biết tại thời điểm đó, vậy thì ta không cần mở rộng nữa, như vậy ta đã cắt bỏ đi một nhánh, giảm được số nghiệm phải tìm rất nhiều.
Điều khó nhất ở đây là phải đánh giá được các nghiệm mở rộng, nếu đánh giá được tốt, thuật toán nhánh cận sẽ chạy nhanh hơn rất nhiều so với vét cạn.

Quy hoạch động

Phương pháp quy hoạch động dùng để giải bài toán tối ưu có bản chất đệ quy, tức là việc tìm phương án tối ưu cho bài toán đó có thể đưa về tìm phương án tối ưu của một số hữu hạn các bài toán con.
Đối với một số bài toán đệ quy, nguyên lý chia để trị (divide and conquer) thường đóng vai trò chủ đạo trong việc thiết kế thuật toán. Để giải quyết một bài toán lớn, ta chia nó thành nhiều bài toán con cùng dạng với nó để có thể giải quyết độc lập.
Điều đó càng thể hiện rõ trong phương án quy hoạch động: Khi không biết phải giải quyết những bài toán con nào, ta sẽ đi giải quyết toàn bộ các bài toán con và lưu trữ những lời giải hay đáp số của chúng với mục đích sử dụng lại theo một sự phối hợp nào đó để giải quyết những bài toán tổng quát hơn.
Và đó chính là điểm khác nhau cơ bản giữa Quy hoạch động và phép phân giải đệ quy, đây cũng chính là nội dung của phương pháp quy hoạch động:
Phép phân giải đệ quy bắt đầu từ bài toán lớn phân ra thành nhiều bài toán con và đi giải từng bài toán con đó. Việc giải từng bài toán con lại đưa về phép phân ra nhiều bài toán nhỏ hơn và lại đi giải các bài toán nhỏ hơn đó bất kể nó đã được giải hay chưa – hay còn gọi là phương pháp theo cấu trúc từ trên xuống (top-down)

Quy hoạch động bắt đầu từ việc giải tất cả các bài toán nhỏ nhất (bài toán cơ sở) để từ đó từng bước giải quyết những bài toán lớn hơn, cho tới khi giải được bài

11

toán lớn nhất (bài toán ban đầu) – hay còn gọi là phương pháp đi từ dưới lên (bottom-up)
Bài toán giải theo phương pháp quy hoạch động gọi là bài toán quy hoạch động. Công thức phối hợp nghiệm của các bài toán con để có nghiệm của bài toán lớn gọi là công thức truy hồi của quy hoạch động.
Tập các bài toán có ngay lời giải để từ đó giải quyết các bài toán lớn hơn gọi là cơ sở quy hoạch động. Không gian lưu trữ lời giải các bài toán con để tìm cách phối hợp chúng gọi là bảng phương án của quy hoạch động.
Để một bài toán có thể áp dụng phương pháp quy hoạch động, ta cần phải xét xem phương pháp đó có thỏa mãn những yêu cầu dưới đây không:
– Bài toán lớn phải phân rã được thành nhiều bài toán con, có sự phối hợp lời

giải của các bài toán con đó để cho ra lời giải của bài toán lớn.
– Vì quy hoạch động là đi giải tất cả các bài toán con, nên nếu không đủ không

gian bộ nhớ vật lý lưu trữ lời giải (bộ nhớ, đĩa, ...) để phối hợp chúng thì phương pháp này không thể thực hiện được.
– Quá trình từ bài toán cơ sở tìm ra lời giải bài toán ban đầu phải qua hữu hạn

bước.

Các bước để chuẩn bị cho một chương trình/ bài toán sử dụng quy hoạch động:

– Giải tất cả các bài toán cơ sở (thông thqờng rất dễ), lqu các lời giải vào bảng

phqơng án.
– Dùng công thức truy hồi phối hợp những lời giải của các bài toán nhỏ đã lqu

trong bảng phqơng án để tìm lời giải của các bài toán lớn hơn rồi lqu chúng vào bảng phqơng án. Cho tới khi bài toán ban đầu tìm đqợc lời giải. – Dựa vào bảng phqơng án, truy vết tìm ra nghiệm tối qu.

Cho tới nay, vẫn chqa có một định lý nào cho biết một cách chính xác những bài toán nào có thể giải quyết bằng phqơng pháp quy hoạch động là

hiệu quả nhất. Tuy nhiên ta có thể xem xét bài toán ấy có thể giải đqợc bằng phqơng pháp này hay không bằng cách trả lời các câu hỏi sau:
1.

—Một nghiệm tối qu của bài toán lớn có phải là sự phối hợp các

nghiệm tối qu của các bài toán con hay không?‖
2.

—Liệu có thể nào lqu trữ đqợc nghiệm các bài toàn con dqới một

hình thức nào đó để phối hợp tìm đqợc nghiệm bài toán lớn?‖
Ta có thể thấy yếu tố bộ nhớ thật sự trở thành vấn đề khi mà kích thqớc bài toán tăng lên mà không có biện pháp lqu lời giải con một cách thích hợp. Đây chính là một điểm yếu của phqơng pháp này. Tuy nhiên vẫn có thể giải

quyết cải tiến bằng cách giảm thiểu bộ nhớ qua xóa kết quả các bài toán con mà chúng ta chắc chắn không dùng đến ở trên.
Trên đây là những đặc điểm chính về phqơng pháp quy hoạch động.
Qua những thuật toán trên, chúng ta có thể thấy phqơng pháp giải chính xác đều giúp đqa ra đqợc kết quả cuối cùng là tối qu. Tuy nhiên sẽ đi kèm với đó là thời gian, đôi khi sẽ lên tới hàm mũ. Do đó ảnh hqởng đến tính thực tế của các phqơng pháp này, gây khó khăn cho việc áp dụng vào các bài toán có kích thqớc lớn. Trong thực tế, đôi khi ta phải chấp nhận những lời giải gần tối qu (tức là chqa phải tối qu nhất) để đổi lấy thời gian tính toán. Vì vậy điều này đã dẫn đến sự ra đời của các phqơng pháp giải gần đúng sẽ đqợc đề cập ở chqơng tiếp theo.
Giải thuật xấp xỉ (phqơng pháp giải gần đúng)
Giải thuật xấp xỉ đqợc đqa ra nhằm khắc phục các hạn của giải thuật chính xác khi có sự bùng nổ tổ hợp, nghĩa là không gian tìm kiếm quá lớn mà nguyên nhân là do kích thqớc dữ liệu đầu vào tăng lên. Mục đích của loại thuật toán này không phải để tìm ra lời giải tối qu chắc chắn mà để tìm ra lời giải gần tối qu nhất nhqng trong thời gian chấp nhận đqợc.
Đã có nhiều giải thuật gần đúng đqợc đqa ra trong những năm gần đây. Và

13

Và chúng ta có thể kể ra các cách tiếp cận sau:
1.

Tìm kiếm Heuristic, trong đó dựa trên phân tích toán học, ngqời ta

đqa ra các quy tắc định hqớng tìm kiếm một lời giải đủ tốt.
2.

Sử dụng các kỹ thuật tìm kiếm cục bộ để tìm lời giải tối qu địa
phqơng.
3.

Tìm lời giải gần đúng nhờ các thuật toán mô phỏng tự nhiên nhq mô

phòng tự nhiên [9, 10, 11] nhq mô phỏng luyện kim (Simulated Annealing SA), giải thuật di truyền (Genetic Algorithm - GA), tối qu bầy đàn (Partical Swarm Optimization - PSO), thuật toán tiến hóa (Evolutionary AlgorithmsEA), ... Những phqơng pháp này là cách tiếp cận Meta-heuristic, đqợc giới thiệu bởi Dorigo năm 1991 [12, 13, 14] đang đqợc nghiên cứu và ứng dụng
rộng rãi cho các bài toán TQ]TH khó [14]
Sau đây tác giả sẽ đi sâu vào hai cách chính đó là Heuristic và Meta-heuristic
Heuristic
Phqơng pháp tham lam (Gready)
Gọi là thuật toán tham lam nhqng thực chất tham lam không đqợc gọi là
thuật toán, mà nó là một kỹ thuật, một phqơng pháp để ta tiến hành giải một bài
toán lập trình. Vậy thì thuật toán tham lam là gì?
Thuật toán tham lam là một thuật toán giải quyết một bài toán theo kiểu
Heuristic để tìm kiếm lựa chọn tối qu ở mỗi bqớc với hy vọng tìm đqợcc tối qu
toàn bộ. Hay nói cách khác, sự lựa chọn tốt nhất ở mỗi bqớc sẽ dẫn tới lời giải tối
qu nhất.
Vậy thì chọn lựa tối qu hóa bằng cách nào?
Giả sử bạn có một hàm cần để tối qu hóa (cực đại hóa hoặc cực tiểu hóa).
Một thuật toán tham lam sẽ thực hiện các lựa chọn tham lam ở mỗi bqớc để đảm
bảo rằng hàm đã cho là tối qu. Thuật toán tham lam chỉ có một lần tính toán lời giải
tối qu với mục đích nó không bao giờ trở lại và đảo ngqợc quyết định.

14

Chẳng hạn áp dụng thuật toán tham lam với bài toán hành trình của ngqời

bán hàng, ta có giải thuật sau: —Ở mỗi bqớc hãy đi đến thành phố gần thành phố
hiện tại nhấtll. Nói chung, giải thuật tham lam có năm thành phần:
– Một tập hợp các ứng viên (candidate), để từ đó tạo ra lời giải.
– Một hàm lựa chọn, để theo đó lựa chọn ứng viên tốt nhất để bổ sung vào lời giải.
– Một hàm khả thi (feasibility), dùng để quyết định nếu một ứng viên có thể đqợc dùng để

xây dựng lời giải.

luanvansieucap

– Khá dễ để tiến hành một thuật toán tham lam cho một bài toán.
– Phân tích thời gian chạy của thuật toán tham lam sẽ dễ dàng hơn kỹ thuật khác (nhq

chia để trị). Với kỹ thuật chia để trị, không rõ ràng liệu kỹ thuật này là nhanh hay chậm. Lý do là ở mỗi mức của đệ quy kích thqớc nhỏ hơn và số lqợng của bài toán con lớn hơn.
– Khó khăn của tham lam là bạn rất vất vả để hiểu chính xác vấn đề. Thậm chí với giải

thuật chính xác rồi, cũng rất khó khăn để chứng minh tại sao nó lại đúng.

Chứng minh một giải thuật tham lam sẽ có cảm giác nhq một nghệ thuật hơn là một môn khoa học, vì nó đòi hỏi rất nhiều sức sáng tạo.
Có một số thuật toán dựa trên tq tqởng của phqơng pháp tham lam thật sự tìm đqợc phqơng án tối qu (chẳng hạn thuật toán Kruscal tìm cây khung cực tiểu), còn lại đa số các thuật toán dựa trên phqơng pháp tham lam thqờng là thuật toán gần đúng, chỉ cho một lời giải xấp xỉ lời giải tối qu.
Meta-heuristic
Thuật toán tiến hóa (Evolutionary Algorithms- EA)
Các thuật toán tiến hóa [1,2,3] là các kỹ thuật tối qu Meta-Heuristics dựa trên nguyên lý của Darwin về sự lựa chọn tự nhiên. Thuật toán bắt đầu với nhóm các cá thể

(gọi là quần thể) trả qua các thao tác (lai ghép, đột biến) tqơng tụ nhq quá trình

sinh sản trong tự nhiên để tạo ra thế hệ các con cháu. Tiếp theo đó, các tqơng tác trên chính các cá thể đó bảo tồn tính di truyền làm cho một số cá thể phù hợp tốt hơn với —môi trqờng‖ và loại bỏ các cá thể xấu đối với —môi trqờng‖. Từ —môi trqờng‖ ở đây đqợc sử dụng nhq một phép ẩn dụ cho ngữ cảnh của hàm mục tiêu đang đqợc tối qu hóa.
Có rất nhiều thuật toán tiến hóa đã đqợc ra đời dựa trên nguyên lý này và đã đqợc áp dụng thành công để giải các bài toán tối qu khákc nhau trong khoa học và kỹ thuật. Nhqng phần lớn trong cách thiết kế của các thuật toán tiến hóa đều chqa tập trung vào giải một bài toán tối qu tại mỗi thời điểm dựa trên một quần thể mà chqa có sự quan tâm đến việc giải quyết nhiều bài toán tối qu khác nhau đồng thời trên cùng một quần thể.
Và trong mảng giải thuật tiến hóa, đã có một số lớp giải thuật đqợc phát triển và áp dụng phổ biến có thể kể đến sau đây:
– Giải thuật di truyền (Genetic Algorithms -GA) – Mô hình mô phỏng qui luật đấu tranh

sinh tồn của tự nhiên.
– Lập trình di truyền (Genetic Programming - GP) – Phqơng pháp máy học nhằm tối qu

quần thể các chqơng trình máy tính để thực hiện một nhiệm vụ tính toán cho trqớc.
– Lập trình tiến hóa (Evolution Programming) – Mô hình mô phỏng các hành vi thích ứng

trong quá trình tiến hóa (tiến hóa kiểu hình).
– Tiến hóa vi phân (Differential Evolution) – Có nhiều đặc điểm tqơng đồng với thuật

toán di truyền (GA) nhqng có các bqớc lai ghép và đột biến có mô tả rõ ràng hơn bằng

các công thức toán.

Giải thuật di truyền (Genetic Algorithms -GA)
Trong số các giải thuật kể trên, thì giải thuật di truyền là một trong những mô hình tính toán phổ biến và thành công nhất trong lĩnh vực tính toán thông minh. Cùng với các kỹ thuật tính toán thông minh khác như tính toán mờ (fuzzy computing), mạng Nơ-ron (neural networks), hệ đa tác tử (multiagent systems), trí

16

tuệ bầy đàn (swarm intelligence), giải thuật di truyền ngày càng phát triển, được áp dụng rộng rãi trong các lĩnh vực của cuộc sống. Tác giả sẽ giới thiệu sơ lược về GA và những bước cơ bản trong GA là gì. Giải thuật di truyền là một kỹ thuật dựa trên cách mô phỏng sự tiến hóa của con người hay của sinh vật nói chung (dựa trên thuyết tiến hóa muôn loài của Darwin – hay nói cách khác là cùng hệ tư tưởng với thuật toán tiến hóa) trong điều kiện quy định sẵn của môi trường. Ý tưởng của giải thuật di truyền để giải một bài toán tối ưu là tìm một tập hợp của những giải pháp, sau đó cho —tiến triển‖ theo hướng chọn lọc để tìm những giải pháp tốt dần hơn. Mục tiêu của giải thuật di truyền là đưa ra lời giải —tốt‖ có thể là tối ưu hay xấp xỉ tối ưu. Cũng như các thuật toán tiến hóa nói chung, hình thành dựa trên quan niệm cho rằng quá tình tiến hóa tự nhiên là quá trình hoàn hảo nhất, hợp lý nhất và tự nó đã mang tính tối ưu. Quá tình tiến hóa tối ưu ở chỗ, thế hệ sau bao giờ cũng tốt hơn (phát triển hơn, hoàn thiện – như đã đề cập ở trên). Các 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 mang những đặc tính của cha mẹ (di truyền), cũng có thể mang những đặc tính 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, dù rằng đột biến xảy ra với xác suất nhỏ hơn nhiều so với hiện tượng di truyền. Các thuật toán tiến hóa tuy có những điểm khác biệt nhưng đều mô phỏng bốn quá trình cơ bản: khởi tạo, lai ghép, đột biến, sinh sản và chọn lọc tự

nhiên.

17

Hình 1-2: Sơ đồ khối cấu trúc thuật toán di truyền

18

Sơ đồ thuật toán GA tổng quát có thể được biểu diễn như sau:

- Kỹ Thuật mã hóa
Mã hóa trong giải thuật di truyền là biểu diễn các nhiễm sắc thể chứa thông tin cho lời giải. Một số cách mã hóa được sử dụng là: mã hóa nhị phân – Binary coding, mã hóa k mức – K-nary coding, mã hóa theo số thực – Real-number coding. Quá trình mã hóa có thể biểu diễn các đầu vào thành các dãy nhiễm sắc thể theo mảng một chiều hoặc nhiều chiều.
Việc lựa chọn phương thức mã hóa tùy thuộc vào bài toán giải quyết. Thông thường hay dùng mã hóa nhị phân. Ví dụ dưới đây mô tả cách mã hóa các số thực thành các bit nhị phân:
VD: Cần mã hóa biến $z \in [x,y] \subseteq$ bằng một tập các bit nhị phân $\{a1,...,aL\} \in \{0,1\}L$

L

Ánh xạ $\Gamma : \{0,1\} \rightarrow [x,y]$ sẽ được xác định như sau:

Tài liệu liên quan

luanvansieucap

Nghiên cứu các thuật toán tạo bóng trong đồ hoạ ba chiều tương tác thời gian thực

70 839 4

Nghiên cứu khoa học " Kết quả bước đầu nghiên cứu về thị trường hàng hóa lâm sản Việt Nam " pot

23 412 0

Đề tài nghiên cứu về thuật toán định tuyến, giao thức định tuyến có sử dụngthuật toán

29 353 1

Nghiên cứu các thuật toán cho dàn ANTEN thích nghi trong hệ thống CDMA

102 901 0

Tiểu luận môn phương pháp nghiên cứu khoa học THUYẾT TIẾN HÓA TRONG CHUỖI THUẬT TOÁN…

41 254 0

Nghiên cứu về thuật toán Hamming

15 528 1

Nghiên cứu về thuật toán định tuyến, giao thức định tuyến có sử dụng thuật toán

29 400 0

NGHIÊN CỨU VỀ THUẬT TOÁN PHÂN LỚP SỬ DỤNG QUÁ TRÌNH HỌC MÁY BÁN GIÁM SÁT, ỨNG DỤNG…

40 846 0

Báo cáo NCKH: ngh g dụng của các phần mềm trong giả hồi quy tương…

49 810 0

Nghiên cứu các thu a quần thể ứng dụng cho bài toán n g

66 422 0