BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC ĐÀ NẴNG

ĐÀO THỊ THẢO SƯƠNG

PHƯƠNG PHÁP QUY HOẠCH ĐỘNG VÀ ỨNG DỤNG DẠY TIN HỌC CHUYÊN TRUNG HỌC PHỔ THÔNG

Chuyên ngành : Khoa học máy tính
Mã số : 60.48.01

TÓM TẮT LUẬN VĂN THẠC SĨ KỸ THUẬT

Đà Nẵng - Năm 2012

Công trình được hoàn thành tại

ĐẠI HỌC ĐÀ NẴNG

Người hướng dẫn khoa học: **PGS.TSKH. TRẦN QUỐC CHIẾN**

Phản biện 1 : **TS. NGUYỄN THANH BÌNH**

Phản biện 2 : **TS. TRẦN THIÊN THÀNH**

Luận văn được bảo vệ tại Hội đồng chấm Luận văn tốt nghiệp thạc sĩ kỹ thuật họp tại Đại học Đà Nẵng vào ngày 19 tháng 01 năm 2013

Có thể tìm hiểu luận văn tại:

- Trung tâm Thông tin - Học liệu, Đại học Đà Nẵng;
- Trung tâm Học liệu, Đại học Đà Nẵng;

MỞ ĐẦU

1. Tính cấp thiết của đề tài

Một trong những tiêu chí để đánh giá chất lượng một trường trung học phổ thông (THPT) đó là số lượng học sinh giỏi của trường so với mặt bằng chung của tỉnh, của cả nước.

Môn Tin học được đưa vào giảng dạy chính thức ở trường THPT từ năm học 2006 -2007 tuy nhiên trong thực tế môn Tin học đã được đưa vào tham gia thi học sinh giỏi cấp tỉnh, cấp quốc gia từ rất lâu: tỉnh đoàn Bình Định tổ chức cuộc thi Tin học trẻ không chuyên lần đầu tiên từ năm 1995, Hội thi Tin học trẻ toàn quốc (tên cũ trước kia là Hội thi tin học trẻ không chuyên toàn quốc) được tổ chức lần đầu tiên vào năm 1995, kỳ thi học sinh giỏi Tin học quốc gia được tổ chức lần đầu tiên vào năm 1995, kỳ thi Olympic Tin học quốc tế (IOI) tổ chức lần đầu vào năm 1989, cuộc thi Olympic Tin học Toàn quốc được tổ chức lần đầu tiên vào năm 1994…).

Chúng ta biết rằng để có kết quả cao trong kỳ thi tuyển chọn học sinh giỏi môn tin học nói chung thì học sinh phải có vốn kiến thức về thuật toán để giải được các bài toán khó (đặc biệt là các thuật toán nâng cao), sau đó học sinh sẽ sử dụng ngôn ngữ lập trình nào đó để lập trình dựa vào thuật toán đã tìm được và giải bài toán theo yêu cầu. Chương trình giảng dạy ở sách giáo khoa của môn Tin học hiện hành trong trường THPT có lượng kiến thức rất hạn chế và vô cùng đơn giản, không đủ cơ sở và không thể là nền tảng để học sinh có thể dựa vào vốn kiến thức đó tham gia một kỳ thi học sinh giỏi cấp tỉnh hay cao hơn. Câu hỏi đặt ra: ***“Làm thế nào để học sinh có thể đạt kết quả cao trong các kỳ thi học sinh giỏi môn Tin học trong trường THPT?”*** yêu cầu các giáo viên giảng dạy môn Tin học trong trường THPT phải suy nghĩ giải quyết.

Quy hoạch động (Dynamic Programming) là một phương pháp rất hiệu quả để giải nhiều bài toán tin học, đặc biệt là những bài toán tối ưu, có một số bài toán sử dụng phương pháp quy hoạch động lại cho hiệu quả cao hơn hơn hẳn so với nhiều phương pháp khác. Số lượng bài toán tin học được giải bằng phương pháp quy hoạch động cũng rất lớn. Số lượng các bài thi có thể áp dụng phương pháp quy hoạch động để giải trong đề thi học sinh giỏi môn Tin học thường rất cao. Vậy ***“Có phải tất cả các bài toán tối ưu đều có thể áp dụng phương pháp quy hoạch động để giải?; “Làm thế nào để nhận dạng được bài toán đó có thể áp dụng phương pháp quy hoạch động để giải?; “Làm thế nào có thể giải một bài toán bằng phương pháp quy hoạch động?”***;…

Vì những lý do trên tôi xin chọn đề tài ***“PHƯƠNG PHÁP QUY HOẠCH ĐỘNG VÀ ỨNG DỤNG DẠY TIN HỌC CHUYÊN TRUNG HỌC PHỔ THÔNG”***

2. Mục tiêu nghiên cứu đề tài

Mục tiêu chính của đề tài là nghiên cứu về phương pháp Quy hoạch động ứng dụng dạy học sinh chuyên tin học khối THPT.

- Giúp cho học sinh chuyên tin học khối THPT thi đạt kết quả ngày càng cao.
- Tạo ra nguồn tài liệu tham khảo về thuật toán hỗ trợ cho học sinh, giáo viên dạy tin học chuyên THPT.

3. Đối tượng và phạm vi nghiên cứu

❖ ***Đối tượng nghiên cứu***

- Phương pháp Quy hoạch động và các bài toán tối ưu.
- Học sinh chuyên tin học khối THPT, giáo viên giảng dạy môn Tin học trong trường THPT.

❖ ***Phạm vi nghiên cứu***

- Phương pháp quy hoạch động, áp dụng phương pháp quy hoạch động để giải các bài toán trong chương trình chuyên tin học THPT.

4. Phương pháp triển khai

a. Phương pháp nghiên cứu tài liệu

- Thu thập, phân tích các tài liệu và thông tin liên quan đến quy hoạch động.
- Lựa chọn một số bài toán bằng phương pháp quy hoạch động trong chương trình tin học chuyên THPT.

b. Phương pháp nghiên cứu thực nghiệm

- Sử dụng phương pháp quy hoạch động bồi dưỡng học sinh giỏi khối 11, 12 tham gia kỳ thi học sinh giỏi cấp tỉnh tại trường THPT Hòa Bình năm học 2011 – 2012.
- Thiết kế các bài toán đã được lựa chọn trong chương trình tin học chuyên THPT bằng phương pháp quy hoạch động.
- Dùng ngôn ngữ lập trình Pascal cài đặt bài toán, chạy thử nghiệm trên một số bộ dữ liệu để đánh giá kết quả.

5. Ý nghĩa khoa học và thực tiễn của đề tài

a. Về mặt lý thuyết:

- Tìm hiểu phương pháp Quy hoạch động.
- Hiểu và vận dụng phương pháp quy hoạch động vào giải các bài toán trong chương trình chuyên tin học THPT đặc biệt là các bài toán tối ưu.

b. Về mặt thực tiễn:

- Tạo nguồn tài liệu tham khảo thuật toán về phương pháp quy hoạch động hỗ trợ cho giáo viên dạy môn tin học, học sinh chuyên tin học THPT.

- Giúp học sinh nhận dạng được bài toán tối ưu nào có thể áp dụng được phương pháp quy hoạch động để giải bài toán.
- Giúp học sinh hiểu và vận dụng được phương pháp quy hoạch động vào giải các bài toán tối ưu để học sinh đạt kết quả cao hơn trong kỳ thi học sinh giỏi các cấp.

6. Bố cục của luận văn:

Nội dung chính của luận văn được chia thành 3 chương như sau:

Chương 1: Cơ sở lý thuyết về quy hoạch động

Chương này giới thiệu những khái niệm cơ bản về quy hoạch động, cách nhận diện xem bài toán tối ưu nào đó có thể áp dụng phương pháp quy hoạch động để giải được hay không? Các bước để giải bài toán bằng phương pháp quy hoạch động.

Chương 2: Một số bài toán quy hoạch động cơ bản dùng để dạy học sinh chuyên THPT.

Chương này giới thiệu một số bài toán tối ưu, phân tích và cách giải bài toán tối ưu đó bằng phương pháp quy hoạch động.

Chương 3: Cài đặt chương trình, kết quả.

Chương này giới thiệu về công cụ lập trình Pascal và dùng Pascal để cài đặt , giải bài toán được giới thiệu ở chương 2 bằng phương pháp quy hoạch động, dùng trình dịch Free Pascal để dịch chương trình, nhận xét kết quả sau khi thực hiện chương trình.

CHƯƠNG 1
CƠ SỞ LÝ THUYẾT VỀ QUY HOẠCH ĐỘNG

1.1. MỘT SỐ KHÁI NIỆM:

1.1.1. Bài toán tối ưu:

a. Khái niệm:

Bài toán tối ưu gồm có 1 hàm f gọi là hàm mục tiêu hay hàm đánh giá; các hàm g_1, g_2, …, g_n cho giá trị logic gọi là hàm ràng buộc. Yêu cầu của bài toán là tìm một cấu hình x thoả mãn tất cả các ràng buộc g_1, g_2, …, g_n:$g_i(x)$ = TRUE ($\forall i: 1 \leq i \leq n$) và x là tốt nhất, theo nghĩa không tồn tại một cấu hình y nào khác thoả mãn các hàm ràng buộc mà f(y) tốt hơn f(x).

Bài toán tối ưu là bài toán thường có nhiều nghiệm chấp nhận được và mỗi nghiệm có một giá trị đánh giá. Mục tiêu đặt ra là tìm nghiệm tối ưu, đó là nghiệm có giá trị đánh giá lớn nhất hoặc nhỏ nhất (tối ưu).

b. Một số ví dụ về bài toán tối ưu:

Ví dụ 1.1:

Trong mặt phẳng tọa độ Oxy tìm tọa độ (x,y) để tổng x + y đạt giá trị lớn nhất mà $x^2 + y^2 \leq 1$.

Ở bài toán trên ta thấy:

Hàm mục tiêu : $x + y \rightarrow \max$

Hàm ràng buộc : $\mathbf{x^2 + y^2 \leq 1}$.

Ví dụ 1.2: Bài toán xếp Ba lô

Có một ba lô có thể chứa tối đa trọng lượng M và có n đồ vật ($n \leq 100$), mỗi đồ vật có trọng lượng w_i và giá trị b_i; M, w_i, b_i là các số nguyên. Hãy chọn và xếp các đồ vật vào ba lô để tổng giá trị của ba lô là lớn nhất.

Với bài toán trên ta thấy:

Hàm mục tiêu: $\sum b_i \to max$, i = 1, 2, …, n

Hàm ràng buộc : $\sum w_i \leq M$, i = 1, 2, …, n .

Tóm lại, bài toán tối ưu rất phong phú, đa dạng, được ứng dụng nhiều trong thực tế nhưng chúng ta cũng cần biết rằng đa số các bài toán tối ưu là không giải được hoặc chưa giải được.

1.1.2. Công thức truy hồi (Hệ thức truy hồi):

Khái niệm: Công thức truy hồi là công thức thể hiện quan hệ giữa các bước trong một bài toán và kết quả của bước sau thường dựa vào kết quả của các bước trước đó. Kết quả của bước cuối cùng là kết quả của bài toán.

1.2. PHƯƠNG PHÁP QUY HOẠCH ĐỘNG:

1.2.1. Phương pháp chia để trị:

"Chia để trị" là việc tách bài toán ban đầu thành các bài toán con độc lập, sau đó giải các bài toán con này rồi tổ hợp dần lời giải từ bài toán con nhỏ nhất đến bài toán ban đầu.

Phương pháp chia để trị là phương pháp thông dụng nhất trong Tin học.

Phương pháp chia để trị thường được áp dụng cho những bài toán có bản chất đệ quy (bài toán P có bản chất đệ quy thì bài toán P có thể được giải bằng lời giải của bài toán P' có dạng giống như P. Tuy nhiên, chúng ta cần lưu ý rằng: P' tuy có dạng giống như P nhưng theo một nghĩa nào đó P' phải nhỏ hơn P, dễ giải hơn P và việc giải nó không cần dùng đến P).

Giải thuật dùng để giải bài toán có bản chất đệ quy gọi là giải thuật đệ quy.

1.2.2. Khái niệm về phương pháp quy hoạch động:

a. Khái niệm:

Phương pháp quy hoạch động (Dynamic Programming) là một kỹ thuật nhằm đơn giản hóa việc tính toán các công thức truy hồi bằng cách lưu toàn bộ hay một phần kết quả tính toán tại mỗi bước trước đó với mục đích sử dụng lại.

Như vậy, Quy hoạch động = Chia để trị + Mảng (lưu lại kết quả).

Phương pháp quy hoạch động do nhà toán học người Mỹ Richard Bellman (1920-1984) phát minh năm 1953. Phương pháp này dùng để giải các bài toán tối ưu có bản chất đệ qui, tức là 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.

Điểm khác nhau cơ bản giữa quy hoạch động và phương pháp đệ quy là :

- Phương pháp đệ quy giải quyết bài toán theo hướng top-down, nghĩa là để giải bài toán ban đầu, ta phải đi giải tất cả các bài toán con của nó. Đây là một phương pháp hay, tuy nhiên phương pháp này sẽ gặp hạn chế về mặt thời gian, tốc độ do phải tính đi tính lại nhiều lần một số bài toán con giống nhau nào đó.
- Phương pháp quy hoạch động sử dụng nguyên lý bottom-up, nghĩa là "đi từ dưới lên". Đầu tiên, ta sẽ phải giải các bài toán con đơn giản nhất, có thể tìm ngay ra nghiệm. Sau đó kết hợp các bài toán con này lại để tìm lời giải cho bài toán lớn hơn và cứ như thế cho đến khi giải được bài toán yêu cầu. Với phương pháp này, mỗi bài toán con sau khi giải xong đều được lưu trữ lại và đem ra sử dụng nếu cần. Do đó tiết kiệm bộ nhớ và cải thiện được tốc độ.

b. Đặc điểm chung của quy hoạch động:

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 toán lớn nhất (bài toán ban đầu).

Quy hoạch động cần phải có bảng phương án

Ý tưởng cơ bản của phương pháp quy hoạch động là tránh tính toán lại các bài toán con đã xét, nói cách khác phương pháp quy hoạch động đã thể hiện sức mạnh của nguyên lý chia để trị đến cao độ.

Tóm lại:

- Quy hoạch động dùng để giải quyết bài toán tối ưu theo nguyên lý “chia để trị” nhưng thực chất là một phương pháp cải tiến hơn của phương pháp giải quyết bài toán theo hướng đệ quy.
- Quy hoạch động làm giảm độ phức tạp, giảm thời gian giải quyết bài
- Quy hoạch động thường tiếp cận theo hướng từ dưới lên (Bottom – up)

1.2.3. Các cách thực hiện phương pháp quy hoạch động

Quy hoạch động thường dùng một trong 2 cách tiếp cận sau:

a. Tiếp cận từ dưới lên (bottom up)

b. Tiếp cận từ trên xuống (top down)

Cách tiếp cận từ dưới lên hiệu quả hơn nên cách tiếp cận từ dưới lên (bottom up) thường được sử dụng nhiều hơn.

1.2.3. Các yêu cầu của một bài toán tối ưu sử dụng được phương pháp quy hoạch động

Một bài toán tối ưu muốn giải được bằng phương pháp quy hoạch động khi bài toán tối ưu đó có các đặc điểm dưới đây:

i. Bài toán lớn phải phân rã được thành nhiều bài toán con, mà sự phối hợp lời giải của các bài toán con đó cho ta lời giải của bài toán lớn.

ii. Vì quy hoạch động là đi giải tất cả các bài toán con nên nếu không đủ không gian vật lý lưu trữ kết quả (bộ nhớ, đĩa …) để phối hợp chúng thì phương pháp quy hoạch động cũng không thể thực hiện được.

iii. Quá trình từ bài bài toán cơ sở tìm ra lời giải bài toán ban đầu phải qua hữu hạn bước.

1.3. CÁC NGUYÊN TẮC CƠ BẢN CỦA QUY HOẠCH ĐỘNG

1.3.1. Nguyên tắc đánh số các giai đoạn từ dưới lên

1.3.2. Nguyên tắc thông số hóa bài toán

1.3.3. Nguyên tác lồng:

1.3.4. Nguyên lý tối ưu Bellman:

Ta biết rằng quy hoạch động thường dùng để giải bài toán tối ưu- bài toán yêu cầu tìm một giải pháp tốt nhất trong các giải pháp có thể tìm được. Cơ sở của quy hoạch động trong bài toán tối ưu là nguyên lý tối ưu Bellman.

Nguyên lý tối ưu Bellman được phát biểu như sau: *"Dãy tối ưu các quyết định trong một quá trình quyết định nhiều giai đoạn có thuộc tính là dù trạng thái và các quyết định ban đầu bất kể như thế nào, những quyết định còn lại phải tạo thành một cách giải quyết tối ưu không phụ thuộc vào trạng thái được sinh ra từ những quyết định ban đầu"*.

Hay nói cách khác: *"Giải pháp tối ưu cho bài toán P cần chứa giải pháp tối ưu cho các bài toán con của P"*.

Do vậy khi giải bài toán theo quy hoạch động nếu dùng một phương pháp gồm nhiều bước tiến hành thì điều kiện cần để giải pháp này tối ưu là nó được xây dựng từ nghiệm tối ưu của những bước trước.

1.4. CÁC BƯỚC GIẢI BÀI TOÁN TỐI ƯU BẰNG QUY HOẠCH ĐỘNG

***Bước* 1: Lập công thức truy hồi**

***Bước* 2: Tổ chức dữ liệu và chương trình**

***Bước* 3: Truy vết, tìm nghiệm của bài toán dựa vào bảng phương án**

1.5. ƯU ĐIỂM VÀ HẠN CHẾ CỦA PHƯƠNG PHÁP QUY HOẠCH ĐỘNG

1.5.1. Ưu điểm:

Tiết kiệm được thời gian thực hiện vì không cần phải tính đi tính lại nhiều lần một số bài toán con giống nhau.

1.5.2. Hạn chế

- Việc tìm công thức truy hồi hoặc tìm cách phân rã bài toán nhiều khi đòi hỏi sự phân tích tổng hợp rất công phu, dễ sai sót, khó nhận ra như thế nào là thích hợp, đòi hỏi nhiều thời gian suy nghĩ. Đồng thời không phải lúc nào kết hợp lời giải của các bài toán con cũng cho kết quả của bài toán lớn hơn.
- Khi bảng lưu trữ đòi hỏi mảng hai, ba chiều … thì khó có thể xử lý dữ liệu với kích cỡ mỗi chiều lớn đến hàng trăm.
- Có những bài toán tối ưu không thể giải được bằng quy hoạch động

Tóm lại:

Không phải lúc nào việc kết hợp các bài toán con cũng cho ta kết quả của bài toán lớn hơn. Hay nói cách khác là việc tìm kiếm "công thức truy hồi" rất khó khăn. Ngoài ra, số lượng các bài toán *con cần lưu trữ có thể rất lớn, không chấp nhận được vì dữ liệu và bộ nhớ máy tính không cho phép.*

Kết chương

Quy hoạch động là một phương pháp hay và hiệu quả, nó có thể giải được hầu hết các bài toán tối ưu. Tuy nhiên, khi giải bài toán theo hướng quy hoạch động, ta cần phải tìm công thức truy hồi thật chính xác và chứng minh độ chính xác tin cậy của nó.

Cho đến nay, vẫn chưa có một định lý nào cho biết chính xác một bài toán tối ưu nào có thể giải quyết hiệu quả bằng quy hoạch động. Tuy nhiên, để biết được bài toán có thể giải bằng phương pháp quy hoạch động hay không, ta có thể tự đặt câu hỏi: " **Một nghiệm tối ưu của bài toán lớn có phải là phối hợp các nghiệm tối ưu của các bài toán con hay không?**" và "**Liệu có thể nào lưu trữ được nghiệm các bài toán con dưới một hình thức nào đó để phối hợp tìm nghiệm của bài toán lớn?**"

CHƯƠNG 2
MỘT SỐ BÀI TOÁN QUY HOẠCH ĐỘNG CƠ BẢN DẠY HỌC SINH CHUYÊN TIN TRUNG HỌC PHỔ THÔNG

Trong chương này chúng tôi xin giới thiệu một số bài toán quy hoạch động cơ bản sử dụng giảng dạy cho học sinh chuyên tin khối THPT.

2.1. TRIỂN KHAI NHỊ THỨC NEWTON $(a+b)^n$

2.1.1. Phát biểu bài toán

Hãy triển khai nhị thức Newton $(a + b)^n$ khi biết giá trị n.

Input: số nguyên dương n.

Output: nhị thức $(a + b)^n$ đã được triển khai.

Ví dụ 2.1: với n = 5:

$(a + b)^5 = a^5 + 5a^4b + 10a^3b^2 + 10a^2b^3 + 5ab^4 + b^5$

2.1.2. Phân tích, xử lý bài toán

Nhị thức Newton được triển khai theo công thức sau:

$$(a+b)^n = \sum_{k=0}^{n} C_n^k a^{n-k} b^k$$

Với $C_n^k = \dfrac{n(n-1)...(n-k+1)}{k(k-1)1} = \dfrac{n!}{k!(n-k)!}$

Như vậy muốn triển khai được nhị thức Newton thì ta phải tính được C_n^k với $0 \le k \le n$.

Công thức cho phép tính tổ hợp chập k của n như sau:

$$C_n^k = \begin{cases} 1 & k=0;\ k=n \\ C_{n-1}^{k-1} + C_{n-1}^{k} & 1 \le k \le n-1 \end{cases}$$

2.1.3. Thuật giải bài toán bằng quy hoạch động

Ta cải tiến thuật toán trên, sử dụng phương pháp quy hoạch động để tính C_n^k như sau:

- Sử dụng mảng C[0..n,0..k] lưu các kết quả C[i, j] đã tính trước đó, trong đó:
- C[i,j] chứa giá trị C_i^j
- C[i,j] được tính:

$$C[i,j] = \begin{cases} 1 & \text{j = 0 hay j = 1} \\ C[i-1, j-1] + C[i-1, j] & \text{j>1} \end{cases}$$

Thuật toán quy hoạch động tính C_n^k như sau:

```
Procedure C[n, k]
begin
  { Sử dụng mảng 2 chiều C[0..n,0..k]}
    for i from 0 to n - k do C[i, 0] = 1 endfor (*)
    for i from 0 to k do C[i, i] = 1 endfor (**)
```

```
  {tính từng cột}
    for j from 1 to k do
          for i from j + 1 to n – k + j do
                C[i, j] = C[ i- 1, j - 1] + C[i - 1, j]  (***)
          endfor
    endfor
return C[n, k]
end
```

Sử dụng thủ tục trên để ta tạo được mảng C[0..n, 0..k]. Để khai triển nhị thức ta ta cần dùng đến C_n^k (1 ≤ k ≤ n - 1), khi đó C_n^k chính là phần tử C[n, k] trong mảng C.

2.1.4. Độ phức tạp thuật toán

Độ phức tạp về thời gian là O(nk).

Độ phức tạp về mặt không gian là O(nk).

2.1.5. Ví dụ minh họa

2.2. DÃY CON CHUNG DÀI NHẤT

2.2.1. Phát biểu bài toán:

Cho hai dãy ký hiệu X và Y, dãy con chung dài nhất của X và Y là dãy các ký hiệu nhận được từ X bằng cách xóa đi một số các phần tử và cũng nhận được từ Y bằng cách xóa đi một số phần tử.

Ví dụ: cho X = ABCDCAE; Y = DACDBA

X =	**A**	B	**C**	**D**	C		**A**	E
Y = D**A**		**C**	**D**		B	**A**		

Dãy con chung dài nhất: ACDA.

2.2.2. Phân tích, xử lý bài toán:

Có nhiều cách để giải quyết bài toán trên: thuật toán vét cạn, đệ quy…

❖ Thuật toán vét cạn:

Độ phức tạp của phương pháp này là $O(m2^n)$ là hàm mũ → không khả thi.

❖ Thuật toán đệ quy:

Thuật toán có độ phức tạp về thời gian $\Omega(2^m)$

2.2.3. Thuật giải bài toán bằng quy hoạch động

- Ta ký hiệu
  - $X_i = x_1x_2\ldots x_i$ được gọi là tiền tố thứ i của X.
  - $Y_j = y_1y_2\ldots y_j$ là tiền tố thứ j của Y.
- Dùng mảng c[0..n, 0..m] để lưu giá trị độ dài của các dãy con chung dài nhất của các cặp tiền tố.
- Gọi c[i, j] là độ dài dãy con chung dài nhất của X_i và Y_j.
- Khi đó độ dài dãy con chung dài nhất của X và Y sẽ là c[n,m].
- Trường hợp đơn giản nhất: độ dài dãy con chung dài nhất của của một dãy rỗng và một dãy bất kỳ luôn bằng 0 do đó c[i, 0] = 0 và c[0, j] = 0 với mọi i, j.
- Vậy ta có:

$$c[i,j] = \begin{cases} 0 & i = 0 \text{ hay } j = 0 \\ c[i-1, j-1] + 1 & i, j > 0, x_i = y_j \\ max(c[i-1, j], c[i, j-1] & i, j > 0, x_i \neq y_j \end{cases}$$

Thuật toán quy hoạch động:

```
Procedure LCS-length(X, Y)
Begin
    n = length (X)
    m = length (Y)
    for  i from 0 to n do c[i, 0] = 0 endfor
    for j from 0 to m do c[0, j] = 0 endfor
```

```
        for i from 1 to n do
            for j from 1 to m do
                if (x_i = y_j) then c[i, j] = c[i - 1, j – 1] + 1
                else
                    c[i, j] = max(c[i - 1, j], c[i, j-1])
                endif
            endfor
        endfor
    return c
    end
```

Truy vết, tìm kết quả

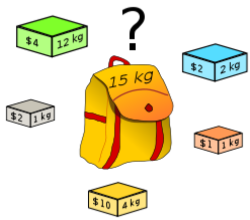
Dãy S chính là dãy kết quả cần tìm, C[n, m] chính là độ dài xâu S đã tìm thấy.

2.2.4. Độ phức tạp thuật toán

Thuật toán tính mảng n x m phần tử bởi hai vòng lặp lồng nhau: độ phức tạp O(nm).

2.2.5. Ví dụ minh họa

2.3. BÀI TOÁN XẾP BA LÔ



2.3.1. Phát biểu bài toán

Có một ba lô có thể chứa tối đa trọng lượng M và có n đồ vật, mỗi đồ vật có trọng lượng w_i và giá trị b_i. M, w_i, b_i là các số nguyên. Hãy chọn và xếp các đồ vật vào ba lô để tổng giá trị của ba lô là lớn nhất.

2.3.2. Phân tích, xử lý bài toán

Có nhiều thuật toán để giải bài toán này: vét cạn, chia để trị…

❖ Thuật toán vét cạn:

Độ phức tạp thuật toán là $O(2^n)$, đây là hàm mũ.

- Thuật toán chia để trị:

  Độ phức tạp của thuật toán $O(2^n)$ là hàm mũ.

2.3.3. Thuật giải bài toán bằng quy hoạch động

- Sử dụng mảng v[0..n,0..M] để lưu trữ lại các giải pháp của các bài toán con.
- Gọi v[i, j] là tổng giá trị lớn nhất của ba lô mà trọng lượng không vượt quá j khi chỉ sử dụng các đồ vật {1, 2, .., i}. Khi đó giá trị lớn nhất khi được chọn trong số n gói với giới hạn trọng lượng M chính là v[n, M].
- Với giới hạn trọng lượng j, việc chọn tối ưu trong số các gói {1, 2, …, i-1, i} để có giá trị lớn nhất sẽ có hai khả năng:
  - Nếu không chọn gói thứ i thì v[i, j] là giá trị lớn nhất có thể bằng cách chọn trong số các gói {1, 2, …, i - 1} với giới hạn trọng lượng là j tức là v[i, j] = v[i – 1, j]
  - Nếu có chọn gói thứ i (tất nhiên chỉ xét tới trường hợp này khi mà $w_i \leq j$) thì v[i, j] = v_i + v[i – 1, j – w_i]
- Vì theo cách xây dựng v[i, j] là giá trị lớn nhất có thể nên v[i, j] sẽ là max trong hai giá trị thu được ở trên tức là :

  v[i, j] = max{ v[i – 1, j], v_i + v[i -1, j – w_i]}
- Ban đầu:
  - v[0, j] = 0 với mọi j.
  - v[i, j] = 0 với mọi i.
- Sau đó v[i, j] sẽ được tính theo v[i-1, j] hoặc v[i - 1, j – w_i].

Thuật toán như sau:

```
Function Balo(n, M)
Begin
```

```
    For i from 0 to n do v[i, 0] = 0 endfor
    For j from 0 to M do v[0, j] = 0 endfor
    For i from 1 to n do
        For j from 1 to M do
            If (wi ≤ j then {có thể sử dụng đồ vật i}
                If (bi + v[i - 1, j – wi] > v[i - 1, j]) then
                 v[i,j] = bi +v[i - 1, j – wi]  {sử dụng đồ vật i}
                else
                    v[i, j] = v[i-1, j]  {không sử dụng đồ vật i}
                endif
            else  {wi > j}
                v[i, j] = v[i-1, j] {không sử dụng đồ vật i}
            endif
        endfor
    endfor
return v[n, M]
end
```

Truy vết, tìm kết quả bài toán.

2.3.4. Độ phức tạp thuật toán

Thuật toán tính mảng n x M phần tử bởi hai vòng lặp lồng nhau: độ phức tạp thời gian là O(nM).

2.3.5. Ví dụ minh họa

2.4. BÀI TOÁN NHÂN TỔ HỢP DÃY MA TRẬN

2.4.1. Phát biểu bài toán:

Có n ma trận M_1, M_2, …, M_n

Với:

M_1 là ma trận kích thước d[1] x d[2]

M_2 là ma trận kích thước d[2] x d[3]

……

M_n là ma trận kích thước d[n] x d[n+1]

Tìm cách kết hợp tích $M_1. M_2. …. M_n$ sao cho thực hiện ít phép nhân nhất.

2.4.2. Phân tích, xử lý bài toán:

- Trước hết, nếu dãy chỉ có một ma trận thì chi phí bằng 0.
- Chi phí để nhân một cặp ma trận có thể tính được ngay: $M_1.M_2$ là d[1] x d[2] x d[3].
- Ta biết rằng phép nhân ma trận không có tính chất giao hoán nhưng có tính kết hợp: (A . B) . C = A . (B . C). Vì thế có thể thực hiện tính tích $M_1. M_2. …. M_n$ bởi nhiều thứ tự kết hợp khác nhau.

Vấn đề đặt ra là cách kết hợp nào thực hiện ít phép nhân nhất.

❖ Thuật toán vét cạn:

Độ phức tạp $\Omega(2^n)$ ➔ **Hàm mũ.**

2.4.3. Thuật giải bài toán bằng quy hoạch động

Nếu dãy chỉ có một ma trận thì chi phí bằng 0.

Chi phí để nhân hai ma trận liên tiếp bất kỳ trong dãy ta có thể tính được ngay và ghi nhận. Sử dụng những thông tin đã ghi nhận để tối ưu hóa những phí tổn nhân bộ ba ma trận liên tiếp… Cứ tiếp tục như vậy cho tới khi ta tính được phí tổn nhân n ma trận liên tiếp.

Ta xây dựng mảng m[1..n,1..n] với m[i,j] là số phép nhân số học tối thiểu cần thực hiện để nhân đoạn ma trận liên tiếp $M_i…M_j$ với $i \leq j$.

Với mỗi m[i,j], $i \leq j$, nên chỉ nửa trên đường chéo chính của mảng m được sử dụng.

Mỗi m[i,j] được tính theo công thức:

$$m_{i,j} = \begin{cases} 0 & i = j \\ \min\limits_{i \le k < j}(m_{i,k} + m_{k+1,j} + d_i d_{k+1} d_{j+1}) & i < j \end{cases} \quad (*)$$

Giải thích công thức trên:

Với i = j: rõ ràng $m_{i,j} = 0$.

Với i < j: $M_i \ldots M_j = (M_i \ldots M_k).(M_{k+1} \ldots M_j)$ có nhiều cách kết hợp với $i \le k < j$.

Với một cách kết hợp (phụ thuộc vào vị trí k), số phép toán tối thiểu phải thực hiện là: **m[i,k]** *(số phép toán nhân số học tối thiểu để tính $M_i \ldots M_k$)* **+ m[k+1,j]** *(số phép nhân số học tối thiểu để tính $M_{k+1} \ldots M_j$)* **+ d[i].d[k+1].d[j+1]** *(số phép nhân số học của tích hai ma trận tích $M_i \ldots M_k$ và $M_{k+1} \ldots M_j$)*. Từ đó suy ra: vì có nhiều cách kết hợp mà ta cần chọn cách kết hợp để có chi phí ít nhất (số phép nhân số học là thấp nhất) nên ta cực tiểu hóa m[i,j] theo công thức (*).

Vậy muốn tính m[i,j] thì ta phải có giá trị m[i,k] và m[k+1,j] với $i \le k < j$.

Mảng m sẽ được xây dựng dần theo từng đường chéo.

Đường chéo s sẽ gồm các phần tử m[i,j] mà j - i = s.

Vậy ta có:

$$m_{i,j} = \begin{cases} 0 & s = 0 \\ \min\,(m_{i,k} + m_{k+1,i+s} + d_i d_{k+1} d_{i+s+1}) & 0 < s < n \end{cases}$$

Xây dựng mảng m mới chỉ cho phép tính số phép nhân tối thiểu (giá trị) của giải pháp.

Ta cần phải xây dựng giải pháp (cách thực hiện các phép nhân ma trận) bằng cách sử dụng bảng phụ g[1..n,1..n] để ghi nhớ chỉ số k cho giải pháp tối ưu mỗi bước.

Thuật toán:

matrixChain(n)

```
begin
    for i from 1 to n do
        for j := 1 to n do
            if i >= j then m[i, j] =0
                else m[i, j] = +∞
            endif
        endfor
    endfor
    for s from 2 to n  do {tính tất cả các đường chéo s}
      for i from 1 to n – s + 1 do{tính một đường chéo s}
            j = i + s - 1
            {tính m_{i,j}=min(m_{i,k} + m_{k+1,j}+ d_i d_{k+1} d_{j+1})
            m[i,j] =  +∞
            for k from i to j - 1 do
                x = m[i,k] +m[k+1,j] + d_i d_{k+1} d_{i+s+1}
                if (x < m[i,j]) then
                    m[i, j]= x
                    g[i, j]=k {ghi nhớ k}
                endif
            endfor
        endfor
    endfor
return m và s
end
```

*** Truy vết kết quả từ các bảng phương án**

2.4.4. Độ phức tạp thuật toán

Thuật toán gồm 3 vòng lặp lồng nhau, mỗi vòng lặp không quá n lần. Vậy độ phức tạp $O(n^3)$.

2.4.5. Ví dụ minh họa

Kết chương:

Bài toán tối ưu thường có nhiều phương pháp giải quyết, tuy nhiên dùng phương pháp quy hoạch động để giải thì giảm đi rất nhiều số lượng thao tác cần thực hiện nên làm giảm độ phức tạp của bài toán.

CHƯƠNG 3
CÀI ĐẶT CHƯƠNG TRÌNH, KẾT QUẢ

3.1. NGÔN NGỮ LẬP TRÌNH PASCAL

Ngôn ngữ lập trình Pascal do Niklaus Wirth phát triển dựa trên Angol năm 1970. Pascal là tên nhà toán học và triết học người Pháp Blaise Pascal.

Pascal là ngôn ngữ đặt biệt thích hợp cho kiểu lập trình cấu trúc. Cho đến nay, Pascal vẫn được dùng để giảng dạy về lập trình trong nhiều trường trung học và đại học trên thế giới. Vì thế chúng tôi chọn ngôn ngữ lập trình Pascal để lập trình cho các bài toán này.

Chúng tôi dùng chương trình dịch Free Pascal để dịch và thực hiện chương trình.

3.2. CÀI ĐẶT CHƯƠNG TRÌNH, KẾT QUẢ THỰC HIỆN

Chương trình được viết bằng ngôn ngữ lập trình Pascal, mỗi bài toán là một chương trình con (thủ tục).

Gồm có 4 chương trình con như sau:

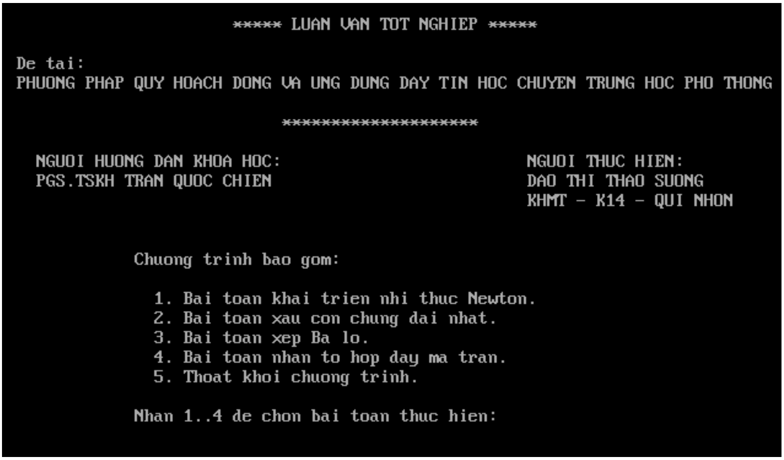
- Thủ tục **Khai_trien_nhi_thuc** dùng để giải bài toán “Khai triển nhị thức Newton”.

- Thủ tục **Xau_con_chung** dùng để giải bài toán "Xâu con chung dài nhất".
- Thủ tục **Xep_ba_lo** giải bài toán "Xếp ba lô".
- Thủ tục **Nhan_ma_trận** giải bài toán "Nhân tổ hợp dãy ma trận".

❖Cài đặt chương trình chính:

Xem phụ lục 2.

❖Giao diện của chương trình chính như sau:



Hình 3.1 Giao diện chương trình chính

Muốn thực hiện bài toán nào thì nhấn số tương ứng (1..4) trong giao diện để thực hiện bài toán đó.

Sau đây chúng tôi xin giới thiệu chương trình của các bài toán:

3.2.1. Bài toán "Triển khai nhị thức Newton"

a. Mô tả dữ liệu

b. Cài đặt

c. Giao diện

3.2.2. Bài toán "Xâu con chung dài nhất"

a. Mô tả dữ liệu

b. Cài đặt

c. Giao diện

3.2.3. Bài toán xếp ba lô

a. Mô tả dữ liệu

b. Cài đặt

c. Giao diện

3.2.4. Nhân ma trận

a. Mô tả dữ liệu

b. Cài đặt

c. Giao diện

KẾT LUẬN

Sau một thời gian nghiên cứu và hoàn thiện luận văn, chúng tôi đã thực hiện được các mục tiêu đề ra như trong thuyết minh đề cương đã được duyệt. Các kết quả đạt được bao gồm:

1. Kết quả đạt được

Đề tài đã đạt được những yêu cầu đã đặt ra về mặt lý thuyết cũng như ứng dụng trong thực tiễn.

Về mặt lý thuyết, đề tài đã trình bày được cơ sở của quy hoạch động: khái niệm, các bước giải bài toán bằng quy hoạch động, cách nhận dạng bài toán có áp dụng được phương pháp quy hoạch động hay không.

Về mặt thực tiễn, luận văn đã giới thiệu được một số bài toán quy hoạch động cơ bản để áp dụng giảng dạy cho học sinh chuyên THPT nhằm giúp cho kết quả thi học sinh giỏi môn Tin học tại trường ngày càng cao.

2. Phạm vi áp dụng

Mặc dù đối tượng nghiên cứu là học sinh giỏi tin học trường THPT Hòa Bình, nhưng đề tài có thể mở rộng và áp dụng bồi dưỡng cho học sinh chuyên tin học ở các trường THPT .

3. Hạn chế:

Số lượng bài toán giới thiệu còn ít.

4. Hướng phát triển

Trong thời gian tới, chúng tôi sẽ tiến hành sưu tầm các bài toán giải được bằng phương pháp quy hoạch động của các kỳ thi chọn HSG cấp tỉnh và cao hơn, phân dạng bài để học sinh sẽ dễ nhận dạng hơn.