ĐẠI HỌC THÁI NGUYÊN

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN&TRUYỀN THÔNG

Đỗ Tuấn Anh

TỔ CHỨC DỮ LIỆU CHO LỚP THUẬT TOÁN CHIA ĐỂ TRỊ VÀ ỨNG DỤNG

Chuyên ngành: Khoa học máy tính

Mã số: 60.48.01

TÓM TẮT LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

Thái Nguyên - 2014

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của bản thân, được xuất phát từ yêu cầu phát sinh trong công việc để hình thành hướng nghiên cứu. Các số liệu có nguồn gốc rõ ràng tuân thủ đúng nguyên tắc và kết quả trình bày trong luận văn được thu thập được trong quá trình nghiên cứu là trung thực chưa từng được ai công bố trước đây.

Thái Nguyên, ngày 19 tháng 5 năm 2014

Học viên thực hiên

Đỗ Tuấn Anh

LỜI CẢM ƠN

Đầu tiên, em xin gửi lời cảm ơn sâu sắc nhất đến cán bộ hướng dẫn khoa học, thầy giáo, PGS.TSKH Nguyễn Xuân Huy, người đã truyền cho em nguồn cảm hứng nghiên cứu khoa học, người đã định hướng cho em đến với lĩnh vực nghiên cứu này.

Em xin bày tỏ lời cảm ơn tới các thầy giáo, cô giáo đã giảng dạy em trong suốt hai năm học qua. Em cũng muốn gửi lời cảm ơn tới những thành viên lớp đã có những góp ý chuyên môn cũng như sự động viên về tinh thần rất đáng trân trọng.

Cuối cùng, em xin gửi lời cảm ơn sâu sắc tới tất cả người thân trong gia đình và những bạn bè em với những động viên dành cho em trong công việc và trong cuộc sống.

Học viên thực hiện luận văn

Đỗ Tuấn Anh

MỤC LỤC

Trang

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

MỞ ĐẦU

.Ngày nay phương pháp này vẫn còn được áp dụng trong nhiều lĩnh vực của đời sống. Đặc biệt, phương pháp này rất hiệu quả khi thiết kế thuật toán giải các bài toán lớn, phức tạp. Với bài toán đầu vào rất lớn ta chia thành những phần nhỏ hơn và tìm lời giải cho các bài toán nhỏ riêng biệt này, rồi sau đó tổng hợp các nghiệm riêng rẽ thành nghiệm bài toán toàn cục.

Trong luận văn này, tôi sẽ tập trung phân tích việc tổ chức dữ liệu cho lớp thuật toán chia để trị và cách đánh giá độ phức tạp đối với các thuật toán chia để trị.Với mục tiêu chính là áp dụng thiết kế thuật toán chia để trị để giải quyết bài toán nhân hai số nguyên lớn, luận văn được trình bày trong 3 chương với bố cục như sau:

Chương 1: Các chiến lược thiết kế thuật toán. Giới thiệu tổng quan về các bước giải bài toán trên máy tính và phân tích đánh giá thời gian thực hiện thuật toán cùng các chiến lược thiết kế thuật toán cơ bản.

Chương 2: Tổ chức dữ liệu cho lớp thuật toán chia để trị. Trình bày ý tưởng, cơ sở khoa học của thuật toán chia để trị và cách thức tổ chức dữ liệu cho thuật toán chia để trị với các bài toán kinh điển.

Chương 3: Ứng dụng thuật toán chia để trị giải bài toán nhân hai số nguyên lớn. Tập trung phân tích các cách tiếp cận giải bài toán nhân hai số nguyên lớn. Từ đó đề xuất thuật toán dựa trên tư tưởng chia để trị để giải quyết và thực nghiệm so sánh với các cách tiếp cận trước đó.

Cuối cùng là **kết luận và hướng phát triển**:Tóm tắtnhững kết quả đạt được, những hạn chế và nêu lên các hướng phát triển trong tương lai.

CHƯƠNG 1. CÁC CHIẾN LƯỢC THIẾT KẾ THUẬT TOÁN

1.1 Các bước cơ bản khi giải bài toán trên máy tính

Một *thuật toán* một thủ tục tính toán được định nghĩa chính xác, mà lấy một giá trị hoặc một tập các giá trị, được gọi là *đầu vào* hay *dữ liệu vào* và tạo ra một giá trị, hoặc một tập các giá trị, và gọi là *đầu ra*. Miêu tả một vấn đề thường được xác định nói chungqua quan hệ đầu vào/đầu ra. Một thuật toán là một dãy bước xác định để chuyển đổi dữ liệu đầu vào thành dữ liệu đầu ra. Chúng ta có thể xem một thuật toán như một công cụ để giải quyết một ***vấn đề tính toán***. Việc trình bày rõ ràng một vấn đề nói chung hình thành mối quan hệ mong muốn đầu vào/đầu ra. Một thuật toán mô tả chính xác một thủ tục tính toán để đạt được mối liên hệ giữa dữ liệu đầu vào và dữ liệu đầu ra.

1.1.1 Xác định bài toán

Việc xác định bài toán tức là phải xác định xem ta phải giải quyết vấn đề gì? với giả thiết nào đã cho và lời giải cần phải đạt những yêu cầu nào. Khác với bài toán thuần tuý toán học chỉ cần xác định rõ giả thiết và kết luận chứ không cần xác định yêu cầu về lời giải, đôi khi những bài toán tin học ứng dụng trong thực tế chỉ cần tìm lời giải tốt tới mức nào đó, thậm chí là tồi ở mức chấp nhận được. Bởi lời giải tốt nhất đòi hỏi quá nhiều thời gian và chi phí.

Input → Process → Output

(Dữ liệu vào →Xử lý →Kết quả ra)

Ví dụ: Khi cài đặt các hàm số phức tạp trên máy tính. Nếu tính bằng cách khai triển chuỗi vô hạn thì độ chính xác cao hơn nhưng thời gian chậm hơn hàng tỉ lần so với phương pháp xấp xỉ. Trên thực tế việc tính toán luôn luôn cho phép chấp nhận một sai số nào đó nên các hàm số trong máy tính đều được tính bằng phương pháp xấp xỉ của giải tích số.

Xác định đúng yêu cầu bài toán là rất quan trọng bởi nó ảnh hưởng tới cách thức giải quyết và chất lượng của lời giải. Một bài toán thực tế thường cho bởi những thông tin khá mơ hồ và hình thức, ta phải phát biểu lại một cách chính xác và chặt chẽ để hiểu đúng bài toán. Trên thực tế, ta nên xét một vài trường hợp cụ thể để thông qua đó hiểu được bài toán rõ hơn và thấy được các thao tác cần phải tiến hành. Đối với những bài toán đơn giản, đôi khi chỉ cần qua ví dụ là ta đã có thể đưa về một bài toán quen thuộc để giải.

1.1.2 Tìm cấu trúc dữ liệu biểu diễn bài toán

Khi giải một bài toán, ta cần phải định nghĩa tập hợp dữ liệu để biểu diễn tình trạng cụ thể. Việc lựa chọn này tuỳ thuộc vào vấn đề cần giải quyết và những thao tác sẽ tiến hành trên dữ liệu vào. Có những thuật toán chỉ thích ứng với một cách tổ chức dữ liệu nhất định, đối với những cách tổ chức dữ liệu khác thì sẽ kém hiệu quả hoặc không thể thực hiện được. Chính vì vậy nên bước xây dựng cấu trúc dữ liệu không thể tách rời bước tìm kiếm thuật toán giải quyết vấn đề.

Các tiêu chuẩn khi lựa chọn cấu trúc dữ liệu:

- Phải biểu diễn được đầy đủ các thông tin nhập và xuất của bài toán
- Phù hợp với các thao tác của thuật toán mà ta lựa chọn để giải quyết bài toán.
- Phải cài đặt được trên máy tính với ngôn ngữlập trình đang sửdụng.

Đối với một sốbài toán, trước khi tổchức dữliệu ta phải viết một đoạn chương trình nhỏ để khảosátxem dữliệu cần lưu trữlớn tới mức độnào.

1.1.3 Xây dựng thuật toán

Thuật toán là một hệ thống chặt chẽ và rõ ràng các quy tắc nhằm xác định một dãy thao tác trên cấu trúc dữ liệu sao cho: Với một bộ dữ liệu vào, sau một số hữu hạn bước thực hiện các thao tác đã chỉ ra, ta đạt được mục tiêu đã định.Các đặc trưng của thuật toán:

1. *Tính đơn định:* Ở mỗi bước của thuật toán, các thao tác phải hết sức rõ ràng, không gây nên sự nhập nhằng, lộn xộn, tuỳ tiện, đa nghĩa. Thực hiện đúng các bước của thuật toán thì với một dữ liệu vào, chỉ cho duy nhất một kết quả ra.
2. *Tính dừng:* Thuật toán không được rơi vào quá trình vô hạn, phải dừng lại và cho kết quả sau một số hữu hạn bước.
3. *Tính đúng:* Sau khi thực hiện tất cả các bước của thuật toán theo đúng quá trình đã định, ta phải được kết quả mong muốn với mọi bộ dữ liệu đầu vào. Kết quả đó được kiểm chứng bằng yêu cầu bài toán.
4. *Tính phổ dụng:* Thuật toán phải dễ sửa đổi để thích ứng được với bất kỳ bài toán nào trong một lớp các bài toán và có thể làm việc trên các dữ liệu khác nhau.
5. *Tính khả thi:*Đối với một bài toán, có thể có nhiều thuật toán nhưng chúng phải cho cùng một output đối với một input. Tuy nhiên chúng có thể khác nhau về hiệu quả. Hiệu quả thời gian là tốc độ xử lý là nhanh hay chậm. Ta có thể đánh giá căn cứ vào số bước thực hiện. Hiệu quả không gian là không gian lưu trữ theo số các đối tượng dùng để ghi nhớ các kết quả (kể cả kết quả trung gian).

Trong Tin học có cả một ngành chuyên đánh giá độ phức tạp của giải thuật, chủ yếu là đánh giá về hiệu quả thời gian. Thực tế sử dụng cho thấy thách thức về không gian lưu trữ có thể giải quyết dễ hơn thách thức về thời gian thực hiện.

1.1.4 Lập trình

Sau khi đã có thuật toán, ta phải tiến hành lập trình thể hiện thuật toán đó. Muốn lập trình đạt hiệu quả cao, cần phải có kỹ thuật lập trình tốt. Kỹ thuật lập trình tốt thể hiện ở kỹ năng viết chương trình, khả năng gỡ rối và thao tác nhanh. Lập trình tốt không phải chỉ cần nắm vững ngôn ngữ lập trình là đủ mà phải biết cách viết chương trình uyển chuyển, phát triển từng bước để chuyển các ý tưởng ra thành chương trình hoàn chỉnh. Kinh nghiệm cho thấy một thuật toán hay nhưng do cài đặt vụng về nên khi chạy lại cho kết quả sai hoặc tốc độ chậm.

Thông thường, chúng ta không nên cụ thể hoá ngay toàn bộ chương trình mà nên tiến hành theo phương pháp tinh chế từng bước (*Stepwiserefinement*):

- Ban đầu, chương trình được thể hiện bằng ngôn ngữ tự nhiên, thể hiện thuật toán với các bước tổng thể, mỗi bước nêu lên một công việc phải thực hiện.
- Một công việc đơn giản hoặc là một đoạn chương trình đã được học thuộc thì ta tiến hành viết mã lệnh ngay bằng ngôn ngữ lập trình.
- Một công việc phức tạp thì ta lại chia ra thành những công việc nhỏ hơn để lại tiếp tục với những công việc nhỏ hơn đó.

Trong quá trình tinh chế từng bước, ta phải đưa ra những biểu diễn dữ liệu. Như vậy cùng với sự tinh chế các công việc, dữ liệu cũng được tinh chế dần, có cấu trúc hơn, thể hiện rõ hơn mối liên hệ giữa các dữ liệu.Phương pháp tinh chếtừng bước là một thểhiện của tưduy giải quyết vấn đềtừtrên xuống, giúpcho người lập trình có được một định hướng thểhiện trong phong cách viết chương trình. Tránhviệc mò mẫm, xoá đi viết lại nhiều lần, biến chương trình thành tờgiấy nháp.

1.1.5 Chạy và kiểm thử

1.1.5.1 Chạy thử và tìm lỗi

Chương trình là do con người viết ra, mà đã là con người thì ai cũng có thể nhầm lẫn. Một chương trình viết xong chưa chắc đã chạy được ngay trên máy tính để cho ra kết quả mong muốn. Kỹ năng tìm lỗi, sửa lỗi, điều chỉnh lại chương trình cũng là một kỹ năng quan trọng của người lập trình. Kỹ năng này chỉ có được bằng kinh nghiệm tìm và sửa chữa lỗi của chính mình.Có ba loại lỗi:

- *Lỗi cú pháp:* Lỗi này hay gặp nhất nhưng lại dễ sửa nhất, chỉ cần nắm vững ngôn ngữ lập trình là đủ. Một người được coi là không biết lập trình nếu không biết sửa lỗi cú pháp.
- *Lỗi cài đặt:* Việc cài đặt thể hiện không đúng thuật toán đã định, đối với lỗi này thì phải xem lại tổng thể chương trình, kết hợp với các chức năng gỡ rối để sửa lại cho đúng.