



**BÁO CÁO CUỐI KỲ**

**\*\*\***

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT

TP. HỒ CHÍ MINH

**CHAPTER 27**

**MÔN HỌC: CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT**

**MÃ SỐ LỚP HP:  DASA230179\_22\_1\_08**

**GVHD: GV.TS. HUỲNH XUÂN PHỤNG**

**THỰC HIỆN: NHÓM 26**

**Đặng Công Tuấn – 21110709**

**Trương Nguyễn Thùy Trang – 21110691**

**Lại Trọng Minh Trường - 21110934**

**Đánh giá đa thức bằng máy tính**

- Trong nhiều ứng dụng, cần phải đánh giá 1 đa thức



- Cách viết lại biểu thức theo nguyên tắc Horner giúp tiết kiệm số phép nhân cần dùng.

- Gần đây, Motzkin [3] đã chỉ ra rằng đối với 1 số đa thức nhất định, vẫn còn cách tốt hơn để thực hiện đánh giá, và khám phá của ông đã truyền cảm hứng cho nghiên cứu dẫn đến bài báo này.

**Cân nhắc sơ bộ**

- Khi đánh giá đa thức bậc 6. Ta thấy cách tiếp cận mới của Motzkin [3] tiết kiệm 2 phép nhân với chi phí là 1 phép cộng bổ sung. Tuy nhiên, phương pháp này đòi hỏi phải giải một phương trình bậc hai, và bậc hai có thể không có bất kỳ gốc thật sự nào. Trên thực tế có nhiều đa thức bậc 6 không thể được xử lý bởi thuật toán của [3]. Và sự sửa đổi chủ yếu của phương pháp đó rõ ràng sẽ được yêu cầu làm cho nó bao hàm tất cả các trường hợp.

- Người ta mong đợi có thể xử lý các đa thức đặc biệt theo cách hiệu quả hơn so với đa thức tổng quát. Vì vậy, chúng ta tự hỏi liệu mọi đa thức bậc 6 có thể được thực hiện với 3 phép nhân hay không. Trong khi cố gắng chứng minh điều này là không thể, tác giả phát hiện ra rằng nó thực sự có thể.

**Trường hợp đặc biệt yn**

- Một số đa thức có các thuộc tính đặc biệt cho phép chúng được đánh giá trong ít bước hơn. Và tất nhiên, một đa thức trong đó tất cả các hệ số của bậc lẻ biến mất, nhìn chung sẽ đòi hỏi ít phép toán hơn

- Ở đây chúng ta chỉ xem xét trường hợp đặc biệt nhất, P = yn. Trong trường hợp này, một trình biên dịch tốt chắc chắn sẽ chuyển đổi điều này bằng cách nào đó thành số phép nhân tối thiểu. Nhưng đó là một vấn đề tổ hợp rất khó để quyết định cách tính toán tốt nhất yn

- Một phương pháp khá nổi tiếng (ví dụ xem Floyd [1, trang 50—51]), mà người ta có thể gọi là binary method, có thể được mô tả như sau:

a) Viết n dưới dạng số trong hệ nhị phân; ví dụ: nếu n là 13 ở dạng thập phân, thì n là 1101 ở dạng nhị phân.

b) Thay mỗi số "1" bằng SX và mỗi số “0” bằng S; ví dụ: 1101 trở thành SXSXSSX.

c) Bỏ SX ở tận cùng bên trái. Chuỗi kết quả có thể được hiểu là: S = "bình phương" và X = "nhân với y." Trong ví dụ, chúng tôi tính y13 với chuỗi SX SSX:

1. Lấy y;

2. Bình phương (cho y2);

3. Nhân với y (cho y3);

4. Bình phương (cho y6);

5. Bình phương (cho y12);

6. Nhân với y (cho y13).

Chúng tôi đã sử dụng 5 phép nhân, trong trường hợp này có thể được hiển thị là tối thiểu

- Một thuật toán khác, được cho là mới, có thể được gọi là Factor Method. Ở đây chúng ta tạo ra các chuỗi S (n, m) như sau:

a) Nếu n không phải là số nguyên tố, S(n, m) = S(n/p,m)S(p, nm/p), trong đó p là thừa số nguyên tố nhỏ nhất của n.

b) Nếu n là số nguyên tố, S(n, m) = S(n - 1, m)Xm.

c) S(l, m) = chuỗi null.

Giải thích: Xm có nghĩa là “nhân với ym”, đó là một kết quả đã được tính toán trước đó. S(n, m) là chuỗi cho phép tính của ymn giả sử rằng ym đã được tính toán. Mục tiêu là tìm S (n, 1).

Ví dụ: S(13, 1) = S(12, 1)X1

= S(6,1)S(2, 6)X1

= S(3,1)S(2, 3)X6X1

= S(2,1)XIX3X6X1

=X1X1X3X6X1

- Cách khác để mô tả thuật toán này là:

a) Phân tích n thành các số nguyên tố, n = p1p2…pr.

b) Nếu n là số nguyên tố, hãy tính (yn-1) \*y

c) Nếu n là hợp số, hãy tính (...((yp1) p2) ...) pr

- Để so sánh hai phương pháp này, Binary Method sử dụng r + s - 1 phép nhân, trong đó r là số nguyên lớn nhất nhỏ hơn hoặc bằng log2 n và s là số thứ nhất trong biểu diễn nhị phân của n. Factor Method sử dụng phép nhân Mn, trong đó:

Diagram

Description automatically generated with low confidence

- Đối với n <= 150, có:

93 trường hợp hai phương pháp bằng nhau,

32 trường hợp Factor method là tốt hơn 1 phép toán,

16 trường hợp Binary method là tốt hơn 1 phép toán,

8 trường hợp Factor method là tốt hơn 2 phép toán,

1 trường hợp (n = 129) Binary method là tốt hơn 2 phép toán.

Các trường hợp nhỏ nhất mà Factor Method vượt trội là n = 15, 27, 30, 31\*, 39, 45. Các trường hợp nhỏ nhất mà Binary Method vượt trội là n = 33, 49, 65, 66, 67, 69.

Nhưng có những trường hợp chúng ta có thể làm tốt hơn cả hai phương pháp. Đối với n <= 70, có 5 trường hợp được biết đến mà điều này là đúng. (Chúng là n = 23, 43, 46, 47, 59).

- Phương pháp thứ ba, là kỹ thuật tốt nhất mà tác giả biết, có thể được gọi là tree method vì nó phát triển một cây. Tree method là một bài tập tốt trong kỹ thuật xử lý danh sách. Bắt đầu ở cấp độ 0 với một nút duy nhất có nhãn 1. Để đến cấp độ tiếp theo khi một cấp độ được hoàn thành, hãy xử lý các nút ở cấp độ trước đó từ trái sang phải. Đối với mỗi nút, hãy thử thêm từng giá trị phía trên nút này, liên tiếp từ trên cùng. Nếu thu được bất kỳ giá trị mới nào, chúng sẽ trở thành các nút phân nhánh sang cấp độ tiếp theo. Các giá trị trùng lặp thu được sẽ bị loại bỏ.

- Ví dụ: chúng ta có thể phát triển ba cấp độ như trong Hình 1. Loại bỏ các giá trị là trùng lặp, chúng ta nhận được Hình 2. Thực hiện các bổ sung thử nghiệm từ trên xuống dưới dẫn đến một cây tốt hơn nhiều so với việc tạo ra chúng từ dưới lên trên. Tuy nhiên, phương pháp từ dưới lên trên thuận tiện hơn nhiều để lập trình và nó có thể được sử dụng nếu các nhánh mới từ nút được gắn từ phải sang trái. Cây bắt đầu trông như trong Hình 3.

Diagram

Description automatically generated

- Chúng tôi chỉ tập trung vào tổng số phép nhân là tiêu chí xuất sắc trong cuộc thảo luận này. Rõ ràng nếu n là một biến không xác định, Phương pháp nhị phân sẽ thích hợp hơn. Trên thực tế, phương pháp đó sẽ khá phù hợp để kết hợp trong phần cứng của máy tính nhị phân, như một toán tử lũy thừa.

- Nếu y là một số thực dấu phẩy động, tất nhiên có một điểm của quy luật hiệu suất giảm dần (point of diminishing returns) khi n dần lớn, vì cuối cùng chúng ta sẽ tốt hơn bằng cách lấy logarit và lũy thừa. Vì Tree method sử dụng r phép nhân ở cấp độ r, có thể ngừng tạo cây ở một mức nhất định; sau đó chúng ta có tập hợp tất cả các giá trị "thú vị" của n. Ví dụ, cây trong Hình 3 trong bài báo cho thấy tất cả n mà người ta biết rằng yn cần ít nhất là 6 phép nhân.

- Mặt khác, factor method có giá trị khi n lớn và một sự ứng dụng đòi hỏi phải tính toán thường xuyên yn