Chuyên đề nâng cao:
Tìm ước của một số bằng kỹ thuật tìm kiếm nhị phân (Phần II)

Tác giả: Phan Thành Hưng (PTNK, ĐHQG TP. HCM)

I. Giới thiệu

Cho bài toán sau:

"Cho một số E, hãy xác định một ước số D của E sao cho $1 < D < E$"

- **Lưu ý:**
  - Những số x thỏa mãn $1 < x < E$ và E chia hết cho x được gọi là **ước không tầm thường (non-trivial divisor)**.
  - Bài toán này có thể được ứng dụng để phân tích thừa số nguyên tố.

II. Thuật toán

1. Thuật toán tối ưu

a. Tận dụng "tối đa" hàm tổng và tích

Vì mỗi ước đều phải đi với cặp: $p \times q = E$ nên ta có $\min(p, q) \leq \sqrt{E}$, nghĩa là nếu E là hợp số, thì luôn tồn tại một ước không tầm thường $\leq \sqrt{E}$. Ta có thể dùng một hàm sigma (tổng) để duyệt x từ 2 đến $\sqrt{E}$ và kiểm tra xem, nếu tới giá trị x đầu tiên là ước của E thì phần dư E khi chia cho x lúc này bằng 0, nên ta có thể cố tình thực hiện một phép chia cho phần dư đó để tạo ra Math ERROR và dừng vòng lặp.

Nhắc lại, phần dư của phép chia X cho Y là $\boxed{X - Y \cdot \text{Int}(X \div Y)}$.

Do đó, ta có thể thực hiện như sau:

$$\sum_{x=2}^{\text{Int}(\sqrt{E})} \left(\frac{0}{E - x\text{Int}(E \div x)}\right)$$

Nếu biểu thức báo lỗi **Math ERROR**, nghĩa là trong đoạn $[2; \sqrt{E}]$ tồn tại ít nhất một ước (cần tìm). Ngay khi báo lỗi xong, thì máy sẽ thoát ra khỏi đa câu lệnh và ta phải thực hiện lại từ đầu. Liệu có thể tận dụng chính điều này để giải quyết phần sau?

b. Tìm ra vị trí của ước

Giả sử ta đã biết nếu E có tồn tại ước thì nó chỉ có thể tồn tại ước trong đoạn $[A; B]$. Từ mục trên, ta có biểu thức kiểm tra xem trong đoạn $[M; B]$ có tồn tại ít nhất một ước nào của E không.

Phan Thành Hưng

$$\lambda_E(M) = \sum_{x=M}^{B} \left(\frac{0}{E - x\text{Int}(E \div x)} \right)$$

Biểu thức trên **báo lỗi Math ERROR** nếu tồn tại ít nhất một ước của E và trả ra 0 (chạy được) nếu không tồn tại.

Dễ thấy, kết cục của hàm $\lambda_E(M)$ có tính chất đơn điệu, vì nếu trong đoạn $[M; B]$ tồn tại ít nhất một ước, thì trong đoạn $[M-1; B]$ chắc chắn cũng phải tồn tại ít nhất một ước như vậy, nghĩa là nếu $\lambda_E(M)$ báo lỗi thì $\lambda_E(M-1)$ cũng báo lỗi. Tương tự, nếu $\lambda_E(M)$ chạy được thì $\lambda_E(M+1)$ cũng chạy được. Kết quả của hàm $\lambda_E(M)$ $\forall M$ sẽ trông như sau: $\underbrace{\dots, X, X, X, \dots, X, X}_{\text{tồn tại ước}}, \underbrace{Y, Y, \dots, Y, Y, Y, \dots}_{\text{không tồn tại ước}}$, với X là báo lỗi, Y là chạy được.

Giả sử ta đã biết ít nhất một ước cần tìm (nếu có) phải nằm trong đoạn $[A; B]$. Ta lấy $M = \text{Int}\left(\frac{A+B+1}{2}\right)$ là giá trị chính giữa và tính giá trị $\lambda_E(M)$.

- Nếu $\lambda_E(M)$ báo lỗi, nghĩa là trong đoạn $[M; B]$ chắc chắn có ước của E. Do đó, ta thu hẹp vùng tìm kiếm lại thành đoạn $[M; B]$ bằng cách gán $A \coloneqq M$.
- Nếu $\lambda_E(M)$ chạy được và không báo lỗi, nghĩa là trong đoạn $[M; B]$ chắc chắn **không** có ước của E, do đó, vùng tìm kiếm được thu hẹp xuống $[A; M-1]$ với hi vọng tìm được ước ở dưới đó, ta gán $B \coloneqq M - 1$.

Cứ như vậy, lặp lại cho tới khi $A \geq B$, nếu A là ước của E thì nó chính là kết quả ta cần tìm. Nếu không, thì chứng tỏ E là số nguyên tố, vì nếu có ước thì toàn bộ đoạn $[A; B]$ ban đầu chắc chắn đã phải tìm được ước rồi.

Thực ra, các bước thực hiện trên chính là kỹ thuật tìm kiếm nhị phân để tìm M lớn nhất mà $\lambda_E(M) = 0$, nghĩa là giá trị M thỏa $\lambda_E(M) = 0$ và $\lambda_E(M+1) = 1$. Nếu trong đoạn $[M; B]$ tồn tại ít nhất một ước của E, mà đoạn $[M+1; B]$ lại không tồn tại bất kì ước nào của E, thì buộc M phải là ước của E, chính là ước ta cần tìm.

c. THUẬT TOÁN CHÍNH

Dựa vào lý thuyết trên, ta dễ dàng hình dung được thuật toán như sau:

- Khởi tạo hai giá trị miền A và B ban đầu lần lượt mang giá trị 1 và $\sqrt{E}$ (vì nếu là hợp số thì luôn tồn tại ước trong khoảng này).
- Tìm giá trị giữa: $M = \text{Int}\left(\frac{A+B+1}{2}\right)$.
- Nếu $\lambda_E(M)$ **báo lỗi** ta gán $A \coloneqq M$ (thu hẹp vùng tìm kiếm lên trên). Ngược lại, nếu $\lambda_E(M)$ **chạy được**, ta gán $B \coloneqq M - 1$ (thu hẹp vùng tìm kiếm xuống dưới).

Lặp lại cho tới khi nào $A \geq B$.

Nếu $A = 1$ thì E là số nguyên tố (vì chỉ có duy nhất ước là 1), nếu không, ước cần tìm chính là A.

Phan Thành Hưng

Sau mỗi lần chạy $\lambda_E(M)$ từ A tới B, vùng tìm kiếm giảm đi một nửa nên tổng số lần lặp là $O\left(\frac{\sqrt{E}}{2}+\frac{\sqrt{E}}{4}+\frac{\sqrt{E}}{8}+\cdots\right)=O\left(\sqrt{E}\right)$. Do vậy, độ phức tạp thuật toán là $O\left(\sqrt{E}\right)$. Trong đó $O\left(\sqrt{E}\right)$ cho toàn bộ thuật toán, và mất $O\left(\log_2\sqrt{E}\right)$ lần bấm đa câu lệnh.

- **Cài đặt vào CASIO**

Biến D được sử dụng nhằm kiểm soát vùng tìm kiếm (các biến A, B). Quy ước rằng nếu $D=0$, ta nâng vùng tìm kiếm lên $[M;B]$, nghĩa là gán $A:=M$. Ngược lại, $D=1$, ta hạ vùng tìm kiếm xuống $[A;M-1]$ bằng cách gán $B:=M-1$.

Ban đầu, ta gán $D=0$ trước khi chạy hàm $\lambda_E(M)$. Nếu $\lambda_E(M)$ báo lỗi, nghĩa là trong đoạn $[M;B]$ có chứa ước cần tìm, nên ta sẽ muốn nâng $A:=M$, chính là rơi vào trường hợp $D=0$ đã gán trước đó. Đồng thời, do khi báo lỗi xong, máy sẽ thoát ra đa câu lệnh và đồng thời hủy thao tác gán tại lệnh đó. Vì vậy, ta hoàn toàn có thể gán luôn $D=1-\lambda_E(M)$, vì nếu $\lambda_E(M)$ báo lỗi, thì D giữ nguyên giá trị 0 trước đó, chính là những gì ta cần. Còn nếu $\lambda_E(M)$ chạy được, nó sẽ trả ra kết quả 0, dẫn đến giá trị mới $D=1-0=1$, cũng chính là giá trị D ta cần nếu không tìm thấy ước để hạ vùng tìm kiếm xuống!

Ngoài ra, ta có thể tận dụng hàm $\mathrm{RanInt\#}(A,B)$ báo lỗi nếu $A\geq B$, ta có thể dùng nó để dừng tự động (không liên quan tới thuật toán).

Cài đặt thuật toán hoàn chỉnh vào máy tính Casio fx-580 VN X:

$$A=0AEMBD\times\mathrm{RanInt\#}\left(A,\mathrm{Int}(B)\right)+DA+(1-D)M:$$
$$B=(1-D)B+D(M-1):$$
$$M=\mathrm{Int}\left(\frac{A+B+1}{2}\right):$$
$$D=0:$$
$$D=1-\sum_{x=M}^{\mathrm{Int}(B)}\left(\frac{0}{E-x\cdot\mathrm{Int}(E\div x)}\right)$$

Giá trị ban đầu: $A=0,\ E=$ dữ liệu đầu vào, $M=1, B=\mathrm{Int}\left(\sqrt{E}\right), D=0$.
Cách dùng: Lặp đa câu lệnh. Nếu gặp **Math ERROR** thì nhấn ◀ hoặc ▶ để quay lại màn hình tính toán. Sau đó bấm CALC (CALC) và bấm = luôn (không chỉnh sửa biến gì hết) để chạy tiếp. Cứ lặp lại tới khi gặp **Argument ERROR** thì dừng. Kiểm tra lại thêm một bước: nếu $A>1$ thì kết quả cần tìm chính là A. Nếu không, E là số nguyên tố.