ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

PHÂN TÍCH VÀ THIẾT KẾ THUẬT TOÁN
CS112.P11.KHTN

BTVN

Sinh viên :
Hoàng Đức Dũng - 23520328
Nguyễn Văn Hồng Thái - 23521418

Giảng viên :
Nguyễn Thanh Sơn

Ngày 7 tháng 11 năm 2024

Mục lục

1 Bài 1

Hãy đưa ra 3 bài toán có ứng dụng 3 kỹ thuật vừa được học: Divide and conquer, Decrease and conquer, Transform and conquer. Giải thích cụ thể áp dụng như thế nào. (Có thể các bài toán lập trình hoặc các ứng dụng thực tế).

1.1 Chia để trị (Divide and Conquer)

Giải thích: Kỹ thuật này chia bài toán thành các bài toán con nhỏ hơn cùng loại, giải quyết chúng độc lập và sau đó kết hợp các kết quả lại.
Ví dụ: Sắp xếp trộn (Merge Sort)

- **Ứng dụng**: Trong thuật toán sắp xếp trộn, một mảng được chia thành hai nửa, sắp xếp chúng độc lập, sau đó trộn lại thành mảng đã sắp xếp. Bước “chia" là tách vấn đề, và bước “trị" là sắp xếp các mảng con.

Ví dụ khác:

- Sắp xếp nhanh (Quick Sort)

1.2 Giảm và trị (Decrease and Conquer)

Giải thích: Kỹ thuật này giải quyết bài toán bằng cách giảm nó xuống một bài toán nhỏ hơn của cùng loại, thường là giải quyết cho đầu vào nhỏ hơn rồi mở rộng lời giải.
Ví dụ: Sắp xếp chèn (Insertion Sort)

- **Ứng dụng**: Thuật toán này xây dựng mảng đã sắp xếp bằng cách lấy từng phần tử một và chèn vào vị trí đúng. Nó giảm kích thước danh sách chưa sắp xếp trong mỗi vòng lặp và giải quyết một bài toán nhỏ hơn.

Ví dụ khác:

- Tìm kiếm nhị phân (Binary Search)
- Thuật toán Euclid tìm ước chung lớn nhất (GCD)

1.3 Biến đổi và trị (Transform and Conquer)

Giải thích: Kỹ thuật này biến đổi bài toán thành dạng đơn giản hơn hoặc thuận tiện hơn, giải bài toán đã biến đổi và sau đó chuyển lời giải về dạng ban đầu.
Ví dụ: Cây tìm kiếm cân bằng (AVL hoặc Red-Black Tree)

- **Ứng dụng**: Trong các cấu trúc dữ liệu như cây AVL, các phép quay được sử dụng để duy trì cấu trúc cân bằng, biến đổi vấn đề tìm kiếm thành một dạng đơn giản hơn bằng cách giữ cho cây cân bằng.

Ví dụ khác:

- Nhân đa thức sử dụng biến đổi Fourier nhanh (FFT)
- Các thuật toán đồ thị (Graph Algorithms)

2 Bài 2:

Cho hai số nguyên x và n ($x \leq 10^{18}, n \leq 10^{18}$), tính tổng:

$$S = x^0 + x^1 + x^2 + \cdots + x^n$$

Ví dụ: Với $x = 5$ và $n = 5$, ta có $S = 3906$.
Yêu cầu:

- Suy nghĩ bài toán trên có mấy cách giải (Gợi ý có ít nhất 2 cách giải) và đối với mỗi cách hãy cho biết ứng dụng những kĩ thuật gì đã học.
- Viết mã giả cho các thuật toán các bạn đã nghĩ ra ở trên

2.1 Brute Force (Tổng trực tiếp)

Kỹ thuật: Phương pháp cơ bản, tính trực tiếp từng lũy thừa x^i với $i \in [0, n]$ và cộng chúng lại.
Độ phức tạp thời gian: $O(n)$
Mã giả:

```
function sum_of_powers(x, n):
    S = 0
    for i = 0 to n:
        S = S + x^i
    return S
```

Giải thích: Phương pháp này sử dụng một vòng lặp đơn giản, tính từng lũy thừa của x và cộng kết quả lại. Đây là cách cơ bản nhưng sẽ không hiệu quả khi n lớn.

2.2 Lũy thừa nhanh (Exponentiation by Squaring)

Kỹ thuật: Chia để trị (Divide and Conquer). Kỹ thuật này giảm số phép nhân cần thiết để tính lũy thừa của x một cách hiệu quả bằng cách sử dụng lũy thừa nhanh.
Độ phức tạp thời gian: $O(\log n)$ cho mỗi x^i, dẫn đến tổng độ phức tạp là $O(n \log n)$.
Mã giả:

```
function fast_power(x, i):
    if i == 0:
        return 1
    y = fast_power(x, i // 2)
    if i is even:
        return y * y
    else:
        return x * y * y

function sum_of_powers_fast(x, n):
    S = 0
    for i = 0 to n:
        S = S + fast_power(x, i)
    return S
```

Giải thích: Phương pháp này sử dụng lũy thừa nhanh để giảm số phép nhân cần thiết khi tính lũy thừa của x.

2.3 Công thức cấp số nhân (Dạng đóng)

Kỹ thuật: Biến đổi và trị (Transform and Conquer). Nếu $x \neq 1$, tổng $S = x^0 + x^1 + \cdots + x^n$ có thể tính bằng công thức cấp số nhân:

$$S = \frac{x^{n+1} - 1}{x - 1}$$

Đây là cách tối ưu vì cung cấp công thức trực tiếp.
Độ phức tạp thời gian: $O(\log n)$ (cần tính x^{n+1} bằng lũy thừa nhanh).
Mã giả:

```
function sum_of_powers_formula(x, n):
    if x == 1:
        return n + 1
    else:
        return (fast_power(x, n + 1) - 1) // (x - 1)
```

Giải thích: Phương pháp này biến bài toán tổng lũy thừa thành một công thức đơn giản dựa trên cấp số nhân.