Nhập môn phân tích độ phức tạp thuật toán-21TN

TRẦN MINH HOÀNG-21120075

Phân tích độ phức tạp của đoạn mã

Đề bài

Cho đoạn mã bên dưới:

```
1 int process(int n){
2     int i = n, m = 1, res = 0;
3     while (i > 0){
4         int j = 1;
5         while (j < m){
6             res = i * j;
7             j = j + ?;
8         }
9         m = m + 1;
10        i = i - @;
11    }
12    return res;
13 }
```

Hãy thay mỗi ký tự `?`, `@` ở dòng code 7, 10 lần lượt bởi các cách sau: $(1, 1)$, (j, j), $(1, j)$, $(j, 1)$. Với mỗi cách, thực hiện đếm số phép gán và ước lượng độ phức tạp tương ứng.

Phân tích chi tiết

Trường hợp 1: (1, 1)

```
1 int process(int n){
2     int i = n, m = 1, res = 0;
3     while (i > 0){
4         int j = 1;
5         while (j < m){
6             res = i * j;
7             j = j + 1;
8         }
9         m = m + 1;
10        i = i - 1;
11    }
12    return res;
13 }
```

Phân tích số phép gán và độ phức tạp:

- Vòng lặp ngoài (while `i > 0`) chạy n lần vì i giảm từ n đến 0 với bước giảm là 1.
- Vòng lặp trong (while `j < m`) chạy số lần phụ thuộc vào giá trị của m. Cụ thể:
  - Đến lần thứ k, vòng lặp trong chạy $k - 1$ lần.

- Tổng số lần vòng lặp trong chạy là:

$$\sum_{k=1}^{n-1} k = \frac{(n-1)n}{2}$$

- Số phép gán `j = j + 1` trong vòng lặp trong:

$$\frac{(n-1)n}{2}$$

- Số phép gán `m = m + 1` thực hiện n lần.
- Số phép gán `i = i - 1` thực hiện n lần.
- Tổng số phép gán là:

$$\frac{(n-1)n}{2} + 2n = O(n^2)$$

Trường hợp 2: (j, j)

```
int process(int n){
    int i = n, m = 1, res = 0;
    while (i > 0){
        int j = 1;
        while (j < m){
            res = i * j;
            j = j + j;
        }
        m = m + 1;
        i = i - j;
    }
    return res;
}
```

Phân tích số phép gán và độ phức tạp:

- Vòng lặp ngoài (while `i > 0`) chạy ít hơn n lần do i giảm nhanh.
- Vòng lặp trong (while `j < m`) chạy $\log(m)$ lần, vì j tăng gấp đôi mỗi lần.
- Số lần thực hiện phép gán `j = j + j` trong vòng lặp trong là $\log(m)$.
- Số lần thực hiện phép gán `m = m + 1` là n lần.
- Số lần thực hiện phép gán `i = i - j` ít hơn n lần, ước lượng là $O(n)$.
- Độ phức tạp tổng quát là $O(n \log(n))$.

Trường hợp 3: (1, j)

```
int process(int n){
    int i = n, m = 1, res = 0;
    while (i > 0){
        int j = 1;
        while (j < m){
            res = i * j;
            j = j + 1;
        }
        m = m + 1;
        i = i - j;
    }
    return res;
}
```

Phân tích số phép gán và độ phức tạp:

- Vòng lặp ngoài (`while i > 0`) chạy ít hơn n lần do i giảm nhanh.
- Vòng lặp trong (`while j < m`) chạy $m - 1$ lần, vì j tăng từ 1 lên $m - 1$.
- Tổng số lần vòng lặp trong chạy là:

$$\sum_{k=1}^{n-1} k = \frac{(n-1)n}{2}$$

- Số lần thực hiện phép gán `j = j + 1` trong vòng lặp trong là:

$$\frac{(n-1)n}{2}$$

- Số lần thực hiện phép gán `m = m + 1` là n lần.
- Số lần thực hiện phép gán `i = i - j` ít hơn n lần, ước lượng là $O(n)$.
- Độ phức tạp tổng quát là $O(n \log(n))$.

Trường hợp 4: (j, 1)

```
int process(int n){
    int i = n, m = 1, res = 0;
    while (i > 0){
        int j = 1;
        while (j < m){
            res = i * j;
            j = j + j;
        }
        m = m + 1;
        i = i - 1;
    }
    return res;
}
```

Phân tích số phép gán và độ phức tạp:

- Vòng lặp ngoài (`while i > 0`) chạy n lần vì i giảm từ n đến 0 với bước giảm là 1.
- Vòng lặp trong (`while j < m`) chạy $\log(m)$ lần do j tăng gấp đôi mỗi lần.
- Số lần thực hiện phép gán `j = j + j` trong vòng lặp trong là $\log(m)$.
- Số lần thực hiện phép gán `m = m + 1` là n lần.
- Số lần thực hiện phép gán `i = i - 1` là n lần.
- Độ phức tạp tổng quát là $O(n \log(n))$.

Tóm tắt

- Trường hợp $(1, 1)$: $O(n^2)$
- Trường hợp (j, j): $O(n \log(n))$
- Trường hợp $(1, j)$: $O(n \log(n))$
- Trường hợp $(j, 1)$: $O(n \log(n))$

Như vậy, các trường hợp (j, j), $(1, j)$, và $(j, 1)$ đều có độ phức tạp thấp hơn $O(n \log(n))$ so với $O(n^2)$ của trường hợp $(1, 1)$.