Interchange Sort

Table of contents

Nghịch thế là một cặp giá trị (a_i, a_j) khi a_i và a_j không thỏa điều kiện sắp thứ tự. Ví dụ nếu mảng một chiều có các phần tử tăng dần mà có một cặp (a_i, a_j) nào đó giảm dần thì cặp đó được gọi là **nghịch thế**.

Idea

Thuật toán Interchange Sort sẽ duyệt qua tất cả các cặp giá trị trong mảng và hoán vị hai giá trị trong một cặp nếu cặp giá trị đó là **nghịch thế**.

Complexity Analysis

Ở mỗi lần lặp theo biến, có n - i lần so sánh (do các số phía trước đã sắp xếp).

Như vậy ta có tổng số lần so sánh là:

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

Trong trường hợp tốt nhất, chuỗi đã được sắp xếp thì chỉ so sánh mà không thực hiện hoán vị.

Còn trong trường hợp xấu nhất khi chuỗi bị đảo ngược thì mỗi lần so sánh đều phải hoán vị.

> Dựa vào số lần thực hiện các toán tử cơ bản như so sánh và hoán vị, ta kết luận được Complexity của một thuật toán bất kỳ. Riêng trường hợp trung bình thì thường khá khó để tính và phải dựa trên sự hiểu biết cùng thông tin về phân bố của dữ liệu đầu vào.

Complexity

Cases	Complexity
Best case	$O(n^2)$
Worst case	$O(n^2)$
Average case	$O(n^2)$

Space Complexity: $O(1)$.

Code

```
void interchangeSort(int* a,int n)
{
    // Do xét cặp nên chỉ xét đến phần tử kế cuối
    for (int i = 0; i < n - 1; i++)
    {
        // So sánh với các phần tử còn lại khác chính nó
        for (int j = i + 1; j < n;j++){

            // Theo thứ tự tăng dần
            if(a[j] > a[i])
            {
                int temp = a[i];
                a[i] = a[j];
```

```
                    a[j] = temp;
                }
            }
        }
    }
```