

Bài 8.6: Thuật toán sắp xếp trộn

- ✓ Các đặc điểm của thuật toán
- ✓ Mô tả thuật toán sắp xếp
- ✓ Mô tả thuật toán trộn
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Các đặc điểm

- Giống như thuật toán sắp xếp nhanh, sắp xếp trộn là một thuật toán chia để trị.
- Thuật toán sắp xếp trộn có độ phức tạp $O(n \log n)$.
- Thuật toán chia mảng cần sắp xếp ra thành hai nửa qua chỉ số phần tử giữa.
- Sau đó tiếp tục đệ quy việc chia tách trên hai nửa vừa tách được đến khi mảng chỉ còn 1 phần tử thì dừng chia tách.
- Lúc này thuật toán tiến hành trộn hai mảng đã sắp xếp lại đến khi nhận được mảng đã sắp xếp hoàn chỉnh.
- Là thuật toán được triển khai trong thư viện của Java.

Mô tả thuật toán sắp xếp

➤ Mã giả và triển khai của thuật toán merge sort:

```
// thuật toán sắp xếp trộn đệ quy
void mergeSort(arr[], first, last) { // first, last: chỉ số p.từ đầu, cuối
    if(first < last) {                // nếu first nhỏ hơn last
        middle = (first + last)/2;    // tìm chỉ số phần tử giữa
        mergeSort(arr, first, middle); // sắp xếp trộn nửa đầu mảng
        mergeSort(arr, middle + 1, last); // sắp xếp trộn nửa sau mảng
        merge(arr, first, middle, last); // trộn hai nửa đã sắp xếp
    }
}

// thuật toán sắp xếp trộn đệ quy, first, last: chỉ số phần tử đầu, cuối
static void mergeSort(int[] arr, int first, int last) { // arr: mảng đầu vào
    if (first < last) {                // nếu first nhỏ hơn last
        int middle = (first + last) / 2; // tìm chỉ số phần tử giữa
        mergeSort(arr, first, middle); // sắp xếp trộn nửa đầu mảng
        mergeSort(arr, first: middle + 1, last); // sắp xếp trộn nửa sau mảng
        merge(arr, first, middle, last); // trộn hai nửa đã sắp xếp
    }
}
```

Mô tả thuật toán trộn

➤ Mã giả của thuật toán merge(phần 1):

```
// thuật toán trộn hai mảng đã sắp xếp tăng dần
void merge(arr[], first, middle, last) {
    // xác định kích thước hai mảng con cần trộn
    size1 = middle - first + 1; // kích thước mảng con trái
    size2 = last - middle; // kích thước mảng con phải
    // tạo mảng con để lưu các phần tử để trộn
    leftArr[] = new T[size1];
    rightArr[] = new T[size2];
    // chép dữ liệu vào mảng con trái
    for(i = 0; i < size1; i++) {
        leftArr[i] = arr[first + i];
    }
    // chép dữ liệu vào mảng con phải
    for(j = 0; j < size2; j++) {
        rightArr[j] = arr[middle + j];
    }
}
```

Mô tả thuật toán trộn

➤ Mã giả của thuật toán merge(phần 2):

```
// tiến hành trộn
i = 0; j = 0; k = first;
while(i < size1 && j < size2) {
    if(leftArr[i] <= rightArr[j]) {
        // gán arr[k] = leftArr[i] sau đó tăng k, i
        arr[k++] = leftArr[i++];
    } else {
        arr[k++] = rightArr[j++]; // gán sau đó tăng k, j
    }
}
// chép các phần tử còn lại của mảng trái vào mảng arr
while(i < size1) {
    arr[k++] = leftArr[i++];
}
// chép các phần tử còn lại của mảng phải vào mảng arr
while(j < size2) {
    arr[k++] = rightArr[j++];
}
```

Mã thật thuật toán trộn

➤ Mã thật của thuật toán merge sort:

```
// thuật toán trộn hai mảng đã sắp xếp tăng dần
static void merge(int[] arr, int first, int middle, int last) {
    // xác định kích thước hai mảng con cần trộn
    int size1 = middle - first + 1; // kích thước mảng con trái
    int size2 = last - middle; // kích thước mảng con phải
    // tạo mảng con để lưu các phần tử để trộn
    int[] leftArr = new int[size1];
    int[] rightArr = new int[size2];
    // chép dữ liệu vào mảng con trái
    for (int i = 0; i < size1; i++) {
        leftArr[i] = arr[first + i];
    }
    // chép dữ liệu vào mảng con phải
    for (int j = 0; j < size2; j++) {
        rightArr[j] = arr[middle + j + 1];
    }
}
```


Mã thật thuật toán trộn

➤ Mã thật của thuật toán merge sort:

```
// tiến hành trộn
int i = 0, j = 0, k = first;
while (i < size1 && j < size2) {
    if (leftArr[i] <= rightArr[j]) {
        // gán arr[k] = leftArr[i] sau đó tăng k, i
        arr[k++] = leftArr[i++];
    } else {
        arr[k++] = rightArr[j++]; // gán sau đó tăng k, j
    }
}
// chép các phần tử còn lại của mảng trái vào mảng arr
while (i < size1) {
    arr[k++] = leftArr[i++];
}
// chép các phần tử còn lại của mảng phải vào mảng arr
while (j < size2) {
    arr[k++] = rightArr[j++];
}
```

Nội dung tiếp theo

Thuật toán sắp xếp nhanh