BÁO CÁO THỰC NGHIỆM CÁC THUẬT TOÁN SẮP XẾP TRONG C++

QUICKSORT, HEAPSORT, MERGESORT AND FUNCTION SORT

Trần Kim Ngọc Ngân | IT003.N21.CTTN

Trần Kim Ngọc Ngân – 22520002
IT003.N21.CTTN – UIT

I. Chuẩn bị

- Một bộ dữ liệu gồm 10 file inp:

+ Mỗi một file gồm một triệu số thực ngẫu nhiên.

+ Dữ liệu của file đầu tiên (“test1.inp) có thứ tự giảm dần.

+ Dữ liệu của file thứ hai (“test2.inp) có thứ tự tăng dần.

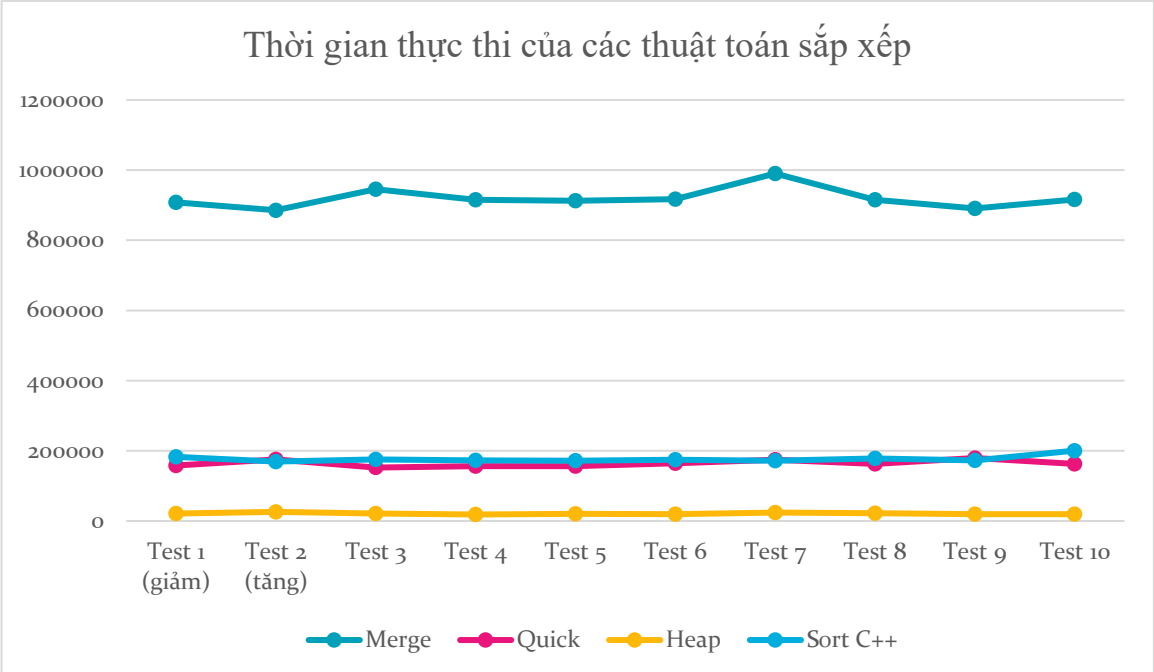
+ Dữ liệu của các file còn lại có thứ tự ngẫu nhiên.

- Các thuật toán sắp xếp: QuickSort, HeapSort, MergeSort và hàm Sort của C++ (các thuật toán được cài đặt tại các file “.cpp”)

- Toàn bộ các file liên quan: https://github.com/KimNgocNgan/Sort_Algorithm

II. Kết quả thử nghiệm

Thuật toán	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
Merge	907569	885578	945450	915272	912450	917089	990070	915234	890229	916093
Quick	158409	175875	152235	156232	156699	164057	174625	162953	179513	162262
Heap	21704	26072	21187	18524	20361	19047	23859	22173	19746	19175
Sort	182717	169360	175375	172887	171747	174957	171827	178335	173027	200422



III. Nhận xét

- Đối với thời gian thực thi: HeapSort có thời gian thực thi ngắn nhất. Thuật toán QuickSort có thời gian thực thi nhanh thứ hai, tiếp đến là hàm Sort của C++. Thuật toán Merge có thời gian thực thi chậm nhất:
- Thời gian sử dụng thuật toán Heap Sort thường là lớn nhất, nếu đối mặt với dữ liệu đã sắp xếp tăng dần thì Merge và Quick sort sẽ tốt hơn, còn đối với dữ liệu đã sắp xếp ngược lại (giảm dần) thì hầu hết các thuật toán đều hoạt động tương đương nhau.
- Với dữ liệu ngẫu nhiên, Merge Sort và Quick Sort đều tương đương nhau, tuy nhiên Heap Sort lại chạy chậm hơn rất nhiều so với hai thuật toán kia.
- Với dữ liệu đã sắp xếp tăng dần, Quick Sort là tốt nhất trong ba thuật toán, với hiệu năng gấp đôi so với Merge Sort và hơn 5 lần so với Heap Sort.
- Với dữ liệu đã sắp xếp giảm dần, Merge Sort và Heap Sort đều tương đương và có hiệu năng tốt hơn Quick Sort gấp gần 5 lần.