Lớp: IT003.M21.KHTN

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Thời gian thực hiện: 03/03/2022 – 16/03/2022

Sinh viên thực hiện: Võ Trần Thu Ngân - 21520069

Nội dung báo cáo: Chạy thử nghiệm các chương trình sắp xếp trên các bộ dữ liệu đầu vào ngẫu nhiên. Ghi lại thời gian thực thi của từng chương trình trên từng tests, lập bảng kết quả, vẽ biểu đồ kết quả và nhận xét. Cụ thể:

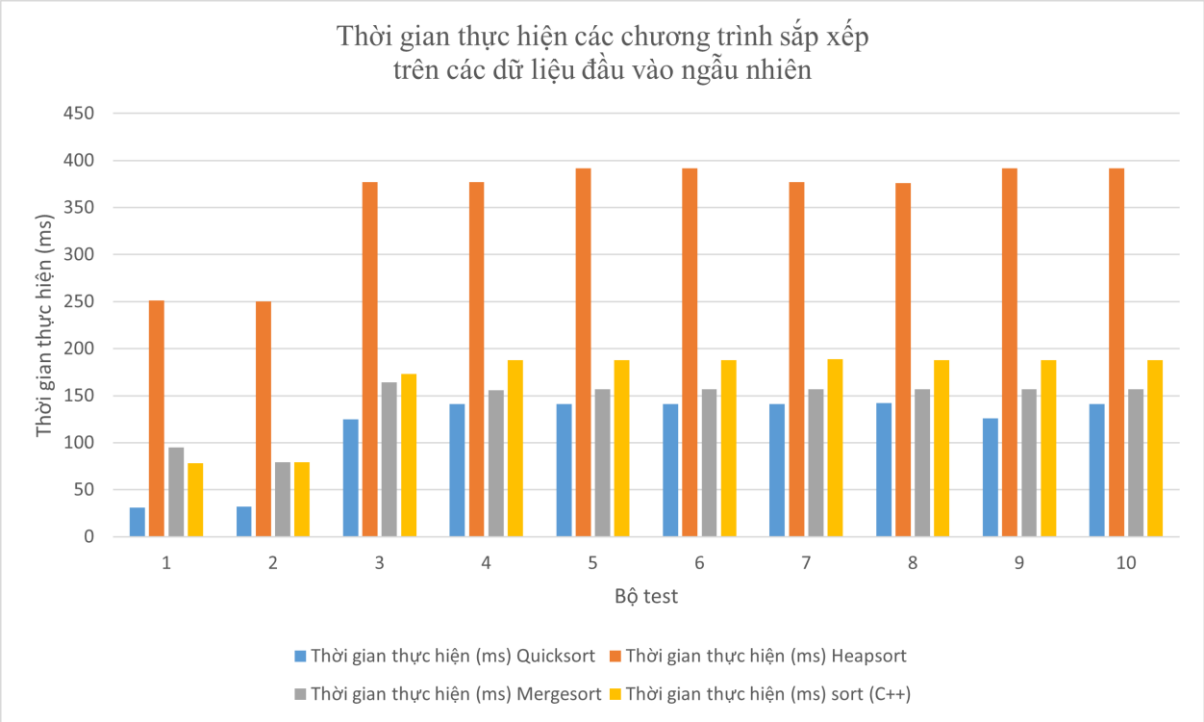
- *Đối tượng thử nghiệm:* chương trình sắp xếp được cài đặt theo thuật toán QuickSort, HeapSort, MergeSort và chương trình gọi hàm std::sort (C++ STL).
- *Dữ liệu thử nghiệm:* gồm 10 tests đầu vào, mỗi test bao gồm 1 triệu số nguyên trong đoạn $[1,10^{18}]$ được sinh ngẫu nhiên. Trong đó, test 1 được sắp xếp tăng dần, test 2 được sắp xếp giảm dần, 8 tests còn lại có thứ tự ngẫu nhiên.
- *Môi trường thử nghiệm:*
    - Hệ điều hành: Windows 11 version 21H2
    - CPU: AMD Ryzen 5 4500U with Radeon Graphics 2.38 GHz
    - RAM: 12GB DDR4
    - Ngôn ngữ sử dụng: C++
    - IDE sử dụng: CodeBlocks 17.12
    - Trình biên dịch sử dụng: GNU GCC (with g++14)

I. Kết quả thử nghiệm

1. Bảng thời gian thực hiện

Dữ liệu	Thời gian thực hiện (ms)			
	Quicksort	**Heapsort**	**Mergesort**	**sort (C++)**
1	31	251	95	78
2	32	250	79	79
3	125	377	164	173
4	141	377	156	188
5	141	392	157	188
6	141	392	157	188
7	141	377	157	189
8	142	376	157	188
9	126	392	157	188
10	141	392	157	188
Trung bình	116.1	357.6	143.6	164.7

2. *Biểu đồ (cột) thời gian thực hiện*



II. *Kết luận:*

- Nhìn chung, QuickSort là thuật toán có thời gian thực thi trung bình nhỏ nhất, ngược lại, HeapSort có thời gian thực thi trung bình lớn nhất (lớn gấp ba lần so với QuickSort).
- Trong các trường hợp tổng quát (chưa được sắp xếp sẵn), các chương trình đều có thời gian thực thi ổn định qua các tests. Cụ thể:
    - QuickSort và MergeSort có thời gian thực thi tương đương nhau. Đây là hai chương trình nhanh nhất.
    - Hàm std::sort có thời gian thực thi khá ngắn, chênh lệch không đáng kể so với MergeSort và QuickSort
    - HeapSort có thời gian thực thi lớn nhất (lớn gấp 2 lần so với hàm sort mặc định của C++).
- Khi thực thi trên hai tests đã được sắp xếp, cả bốn chương trình đều đạt được thời gian thực thi rất ngắn so với các trường hợp tổng quát, đặc biệt là QuickSort (chỉ bằng 1/4 so với trường hợp tổng quát, và bằng 1/3 so với MergeSort và std::sort trong cùng trường hợp).

Vậy: Trong các trường hợp tổng quát, ta nên sử dụng QuickSort hoặc MergeSort để đạt được hiệu quả về thời gian. Khi gặp trường hợp đặc biệt, độ phức tạp của QuickSort có thể lên đến $O(N^2)$ nhưng ta có thể tránh được bằng cách chọn khóa ngẫu nhiên. Hàm sort mặc định của C++ nên được cân nhắc sử dụng vì nó đảm bảo được hiệu quả thời gian thực thi và cũng tiết kiệm được thời gian cài đặt chương trình.

III. *Thông tin chi tiết – link github*

https://github.com/nganngants/IT003_Homeworks/tree/main/sorting