Lớp: KHTN2021

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Thời gian thực hiện: 011/03 – 16/03/2022

Sinh viên thực hiện: Phan Trọng Nhân

Mã số sinh viên: 21522407

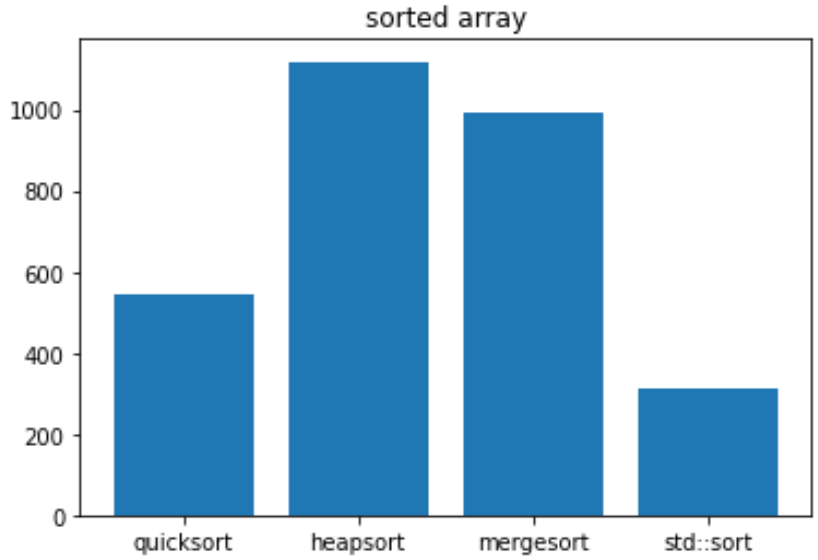
Nội dung báo cáo: Khảo sát độ hiệu quả của một số thuật toán sắp xếp có độ phức tạp O(NlogN) bằng ngôn ngữ C++

I. Kết quả thử nghiệm

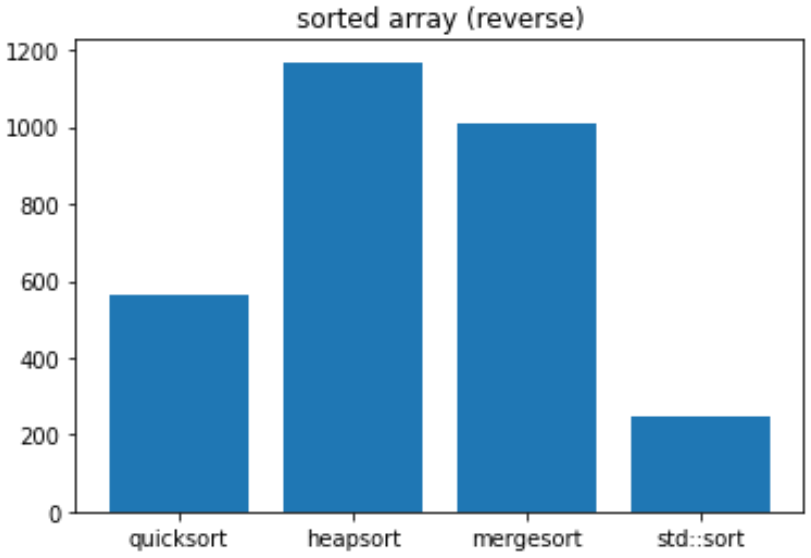
1. Bảng thời gian thực hiện

Dữ liệu	Thời gian thực hiện (ms)			
	Quicksort	**Heapsort**	**Mergesort**	**sort (C++)**
1	545	1118	993	310
2	562	1169	1011	247
3	989	1494	1440	568
4	959	1476	1447	543
5	1000	1650	1448	542
6	969	1502	1756	594
7	946	1521	1676	598
8	947	1491	1429	635
9	1087	1557	1544	628
10	1166	1552	1578	539
Trung bình	917	1453	1432,2	520,4

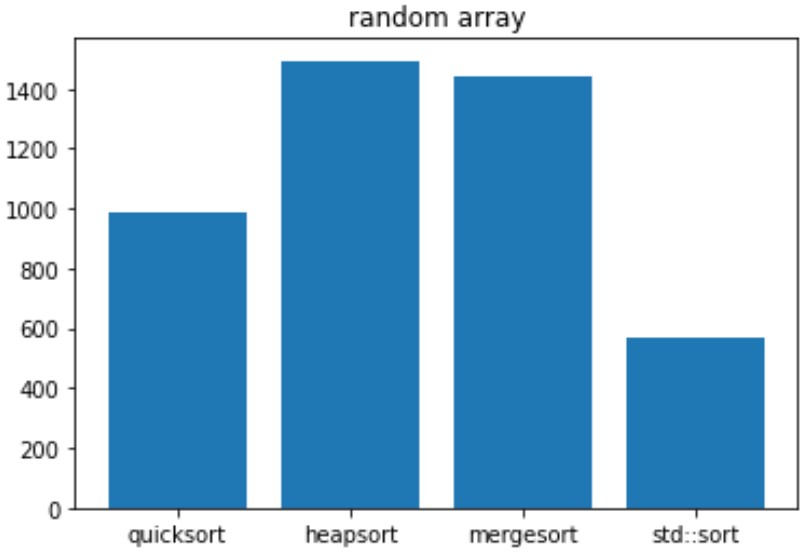
2. Biểu đồ (cột) thời gian thực hiện (thời gian thực hiện tính bằng ms)



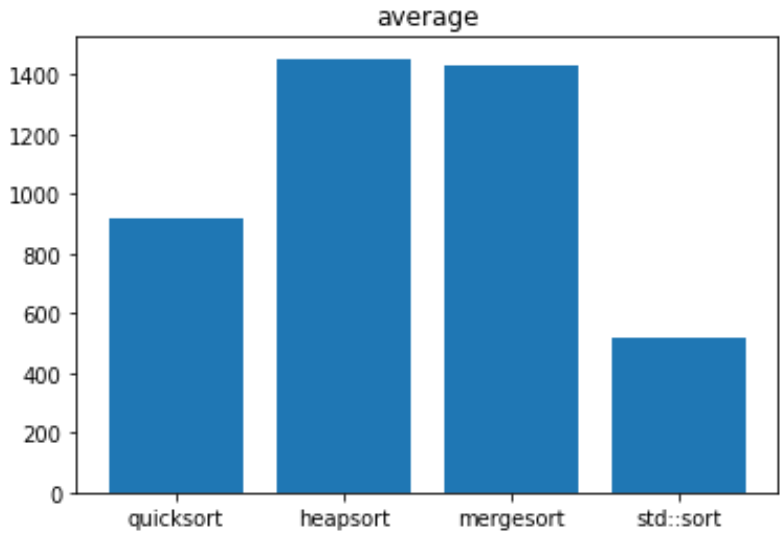
Minh họa thời gian thực hiện với dữ liệu đã sắp xếp (tăng dần)



Minh họa thời gian thực hiện với dữ liệu đã sắp xếp (giảm dần)



Minh họa thời gian thực hiện với dữ liệu ngẫu nhiên



Minh họa thời gian thực hiện trung bình các dữ liệu

II. Nhận xét và giải thích

- **Heap sort** chậm nhất, nhưng ưu điểm của nó là không yêu cầu đệ quy/tạo thêm nhiều mảng.
- **Merge sort** nhanh hơn heap sort một chút, nhưng nó yêu cầu gấp đôi bộ nhớ để tạo mảng phụ. Một ưu điểm nữa là nó hạn chế được việc đọc/ghi từ bộ nhớ.
- **Quick sort** nhanh trong đa số các trường hợp, và yêu cầu ít bộ nhớ. Tuy nhiên, một nhược điểm của nó là đọc từ bộ nhớ nhiều lần. Khi dữ liệu cực lớn và trong các thiết bị mà đọc/ghi từ bộ nhớ tốn nhiều thời gian thì merge sort sẽ nhanh hơn nhiều.
- **std::sort** trong C++ dùng quicksort hoặc introsort, thực tế, nó chỉ giúp thuật toán ổn định hơn, chứ không giúp thuật chạy nhanh hơn.
- Trong khảo sát này, std::sort và heap sort đáng lẽ nên chạy chậm hơn một chút, lí do nó chạy nhanh hơn trong khảo sát là vì code std::sort, heap sort dùng để khảo sát sử dụng công cụ sắp xếp được dựng sẵn trong ngôn ngữ, nên sử dụng được các ưu điểm của ngôn ngữ.
- Ngoài ra, còn rất nhiều các yếu tố khác ảnh hưởng tới thời gian chạy như bộ nhớ, cách đọc ghi file,... tuy nhiên không thuộc phạm trù môn học nên sẽ không được đề cập.

III. Kết luận: Trong khảo sát, std::sort nhanh nhất, và nên luôn được dùng nếu không có ràng buộc nào khác trừ thời gian. Nhưng khi có những ràng buộc, những thuật toán khác sẽ phát huy ưu điểm của nó: Khi bị hạn chế về bộ nhớ thì nên dùng heap sort, khi việc đọc/ghi từ bộ nhớ tốn nhiều thời gian thì dùng merge sort. Std::sort cũng bị hạn chế khi nó chỉ dùng được trong C++.

IV. Link Github: https://github.com/PTN407/sort