Lớp: ANTN 2021

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

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

Sinh viên thực hiện: **Liêu Minh Nhật**

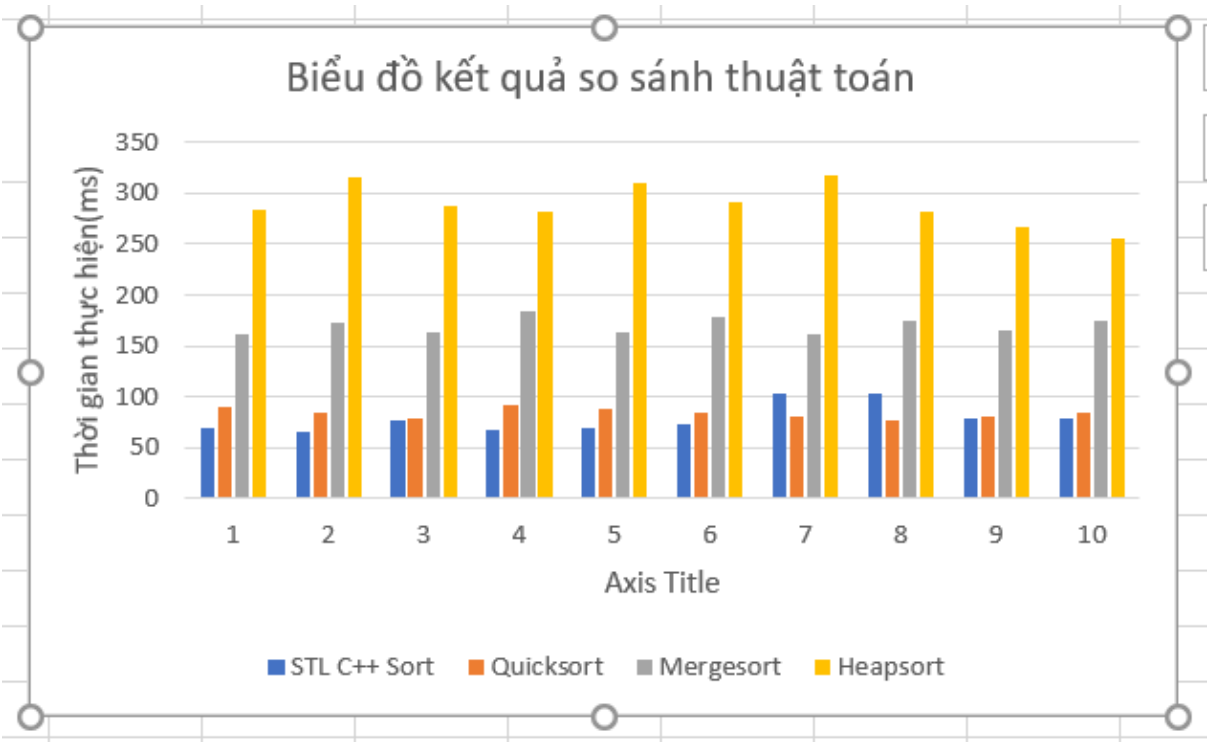
Nội dung báo cáo: Kiểm nghiệm và so sánh thời gian thực hiện của các thuật toán sắp xếp Quicksort, STL C++ sort, Mergesort và Heapsort.

I. Kết quả thử nghiệm

1. Bảng thời gian thực hiện[1]

Dữ liệu	Thời gian thực hiện (ms)			
	STL C++ Sort	Quicksort	Mergesort	Heapsort
1	70	90	161	284
2	66	84	172	315
3	76	78	164	288
4	67	91	185	281
5	70	88	164	310
6	73	85	178	292
7	104	80	161	318
8	103	77	175	281
9	79	80	166	266
10	79	84	174	255
Trung bình				

2. Biểu đồ (cột) thời gian thực hiện



II. Kết luận:

[1] Số liệu chỉ mang tính minh họa

- Nhìn chung theo kết quả thực nghiệm thì ta thấy rằng STL Sort chạy tốt hơn 3 loại Sort còn lại ở trong cả 10 trường hợp. Nhanh thứ 2 là Quicksort, rồi đến Mergesort và cuối cùng là Heapsort
- Ta có thể nói STL Sort là phiên bản tốt hơn của Quicksort bởi theo https://www.geeksforgeeks.org/sort-algorithms-the-c-standard-template-library-stl/; STL sử dụng kết hợp cả bộ 3 Insertionsort, Mergesort và Quicksort. Chủ yếu là nó hoạt động như Quicksort nhưng nếu Quicksort partition không thuận lợi và dẫn đến việc là hoạt động quá nlogn thì nó sẽ chuyển qua Heapsort khi duyệt trên mảng có kích thước bé, và Insertionsort khi mảng có kích thước lớn.
- Quicksort và Mergesort đều dùng tư tưởng chia để trị và đều có độ phức tạp thời gian là O(nlogn). Vậy tại sao Quicksort lại nhanh hơn? Điều đó có thể giải thích là do Quicksort không cần tốn thêm bộ nhớ để trộn các dãy lại như Mergesort và đặc biệt là cache performance của Quicksort tốt hơn.
- Điều thú vị là nếu cố tình chọn test chết thì Quicksort có thể có độ phức tạp $O(n^2)$ nhưng ta có thể tránh bằng cách chọn random pivot.

III. Thông tin chi tiết – link github, trong repo gibub cần có

1. Báo cáo
2. Mã nguồn
3. Dữ liệu thử nghiệm

Link Github: https://github.com/LMNhat/21520377-ATTN-SortingReport

Cảm ơn thầy đã đọc!