Lớp: TTNT2025

BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Sinh viên thực hiện: Phạm Thành Trung (MSSV 25521970)

Nội dung báo cáo:

Báo cáo quá trình thử nghiệm và so sánh thời gian chạy của 3 thuật toán sắp xếp tự cài đặt bằng ngôn ngữ Python (QuickSort, HeapSort, MergeSort) và hàm sort() có sẵn của thư viện Numpy. Thử nghiệm được tiến hành trên 10 dãy dữ liệu ngẫu nhiên, mỗi dãy gồm 1.000.000 phần tử (gồm số nguyên và số thực) với các trật tự khác nhau (tăng dần, giảm dần, ngẫu nhiên).

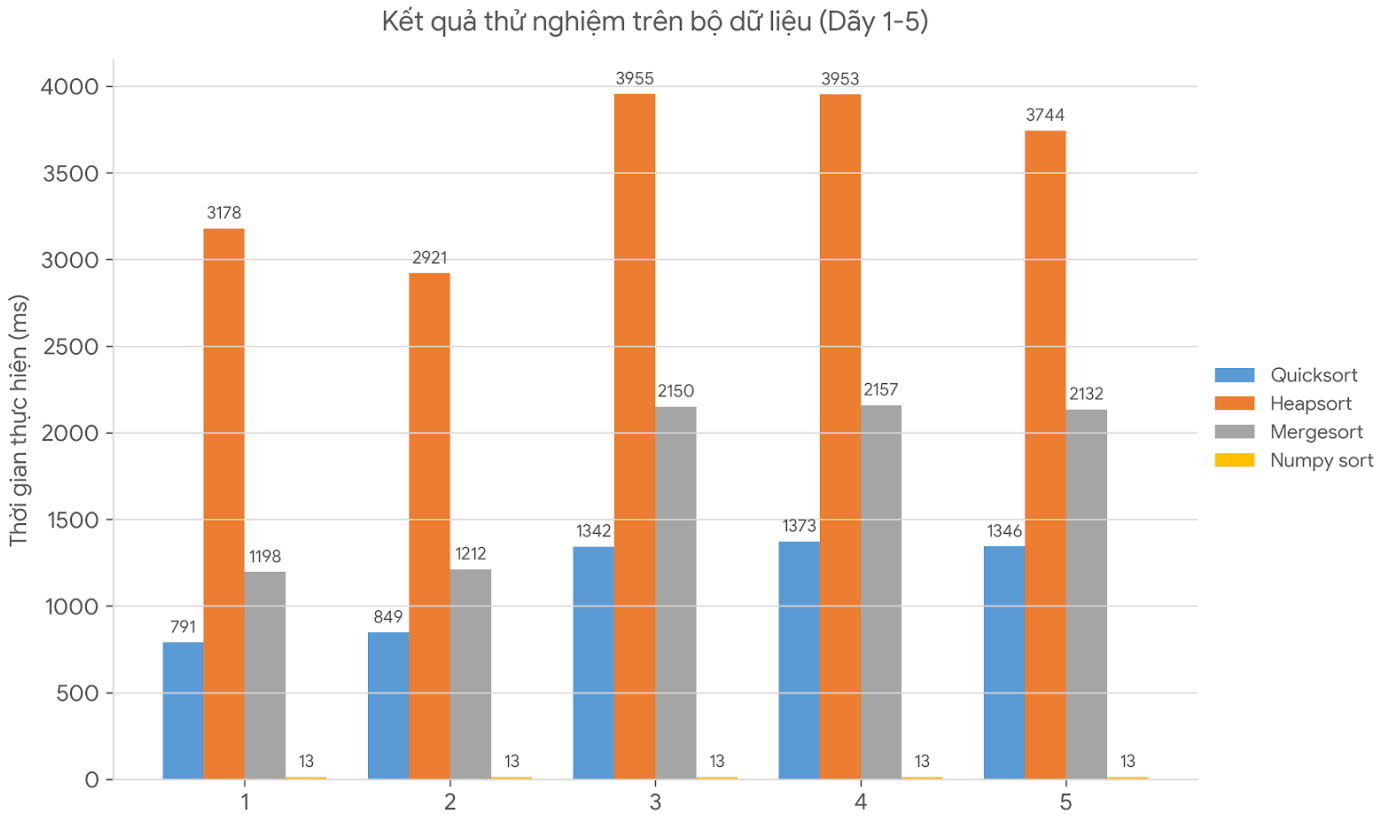
I. Kết quả thử nghiệm

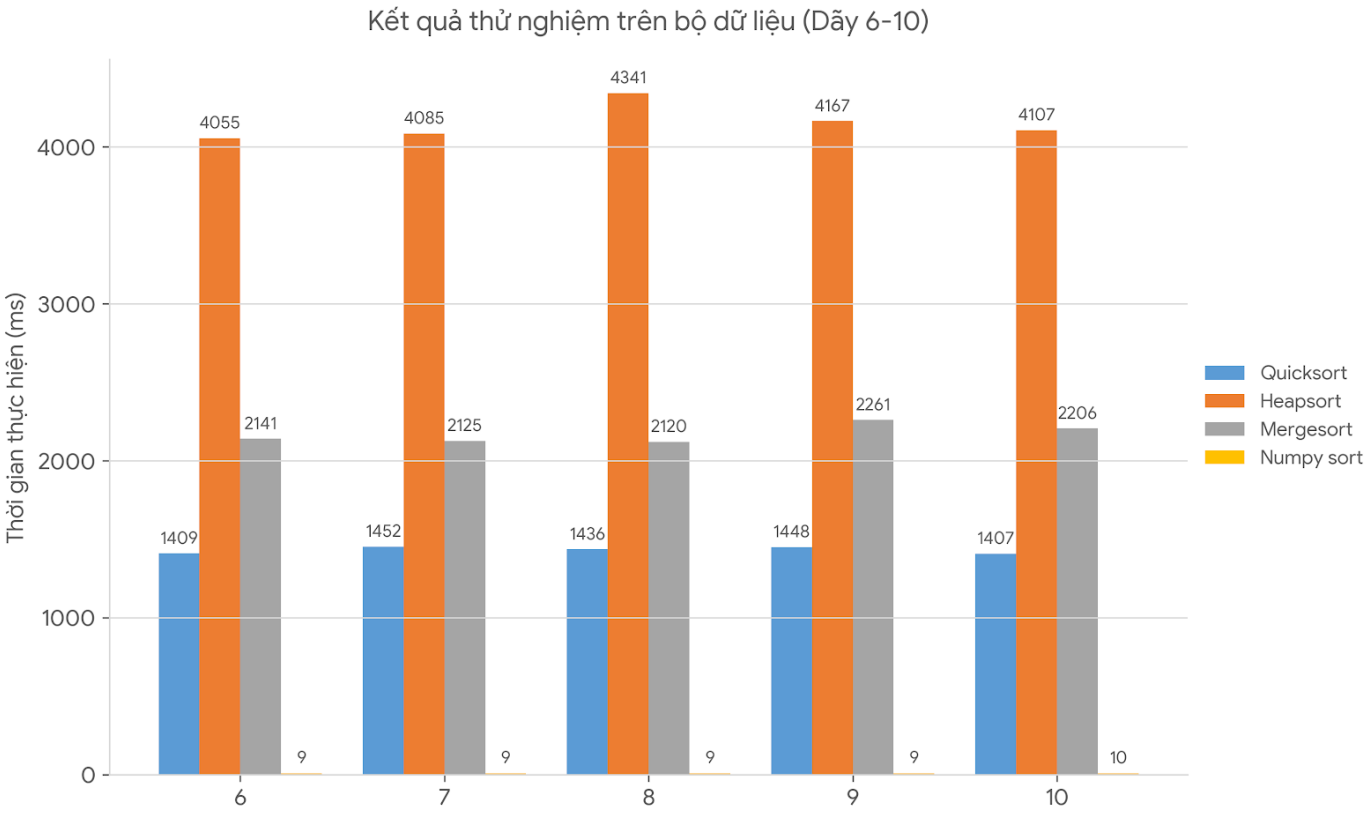
1. Bảng thời gian thực hiện[1]

Dữ liệu	Thời gian thực hiện (ms)				
	Quicksort	**Heapsort**	**Mergesort**	**sort (C++)**	**sort (Numpy)**
1	790.93	3177.81	1198.10		13.10
2	848.64	2921.27	1211.63		13.39
3	1342.41	3954.87	2149.69		13.26
4	1372.52	3953.19	2157.04		13.16
5	1345.95	3744.15	2132.09		13.21
6	1409.09	4055.15	2140.52		9.19
7	1451.82	4084.76	2424.58		9.32

8	1435.56	4340.79	2119.61		8.86
9	1447.89	4166.78	2261.24		8.89
10	1407.33	4106.98	2206.24		9.73
Trung bình	1285.21	3850.58	1970.07		11.21

2. Biểu đồ (cột) thời gian thực hiện





II. Kết luận:

Dựa trên bảng số liệu và biểu đồ thực nghiệm, ta có thể rút ra các nhận xét sau:

- Đối với các thuật toán tự cài đặt (QuickSort, MergeSort, HeapSort): QuickSort thể hiện tốc độ ấn tượng nhất với thời gian trung bình khoảng 1285 ms. Tiếp theo là MergeSort (~1970 ms) và chậm nhất là HeapSort (~3851 ms). Đặc biệt, QuickSort xử lý các mảng đã có thứ tự (dãy 1 và 2) chỉ tốn khoảng 790-850ms do phần tử pivot trong trường hợp này chia mảng khá tối ưu.

- Đối với Numpy Sort: Hàm sort() của Numpy bỏ xa hoàn toàn các thuật toán thủ công với thời gian trung bình chỉ vỏn vẹn 11.21 ms. Sự chênh lệch đáng kể này là do các hàm cốt lõi của Numpy được biên dịch bằng C/C++, cho phép giao tiếp sát với phần cứng máy tính. Thêm vào đó, Numpy lưu trữ dữ liệu thành các khối liên tục giúp tối đa hóa tốc độ đọc của bộ nhớ đệm CPU, đồng thời sử dụng thuật toán lai IntroSort siêu việt.

- Về kiểu dữ liệu: Quá trình sắp xếp mảng *float* bằng Numpy trên hệ thống thử nghiệm cho tốc độ nhỉnh hơn một chút so với mảng *int.* Trong khi đó, với các thuật

toán Python thủ công (như HeapSort), việc hoán đổi và tính toán trên số thực lại mất nhiều thời gian hơn đáng kể so với số nguyên.

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

Link github: https://github.com/phthtrrr/Sorting_Algorithms_Report