BÁO CÁO KẾT QUẢ THỬ NGHIỆM

Sinh viên thực hiện: Dương Đăng Quang
Lớp: KHTN2025 **MSSV:** 25521516

Link Github: https://github.com/NotAvailableRightNow/Sorting-Algorithms-Report.git

Nội dung báo cáo

Báo cáo trình bày kết quả nghiên cứu và đánh giá hiệu năng của các thuật toán sắp xếp: **QuickSort**, **HeapSort**, **MergeSort** và **NumPy Sort**. Thử nghiệm được thực hiện trên bộ dữ liệu gồm 10 dãy số (kích thước 10^6 phần tử). Dãy thứ nhất đã có thứ tự tăng dần, dãy thứ hai có thứ tự giảm dần, 8 dãy còn lại trật tự ngẫu nhiên (5 dãy số thực, 5 dãy số nguyên). Kết quả thử nghiệm được minh họa chi tiết qua bảng số liệu và biểu đồ so sánh thời gian thực thi, từ đó rút ra nhận xét về tính ổn định và ưu nhược điểm của từng thuật toán trong thực tế.

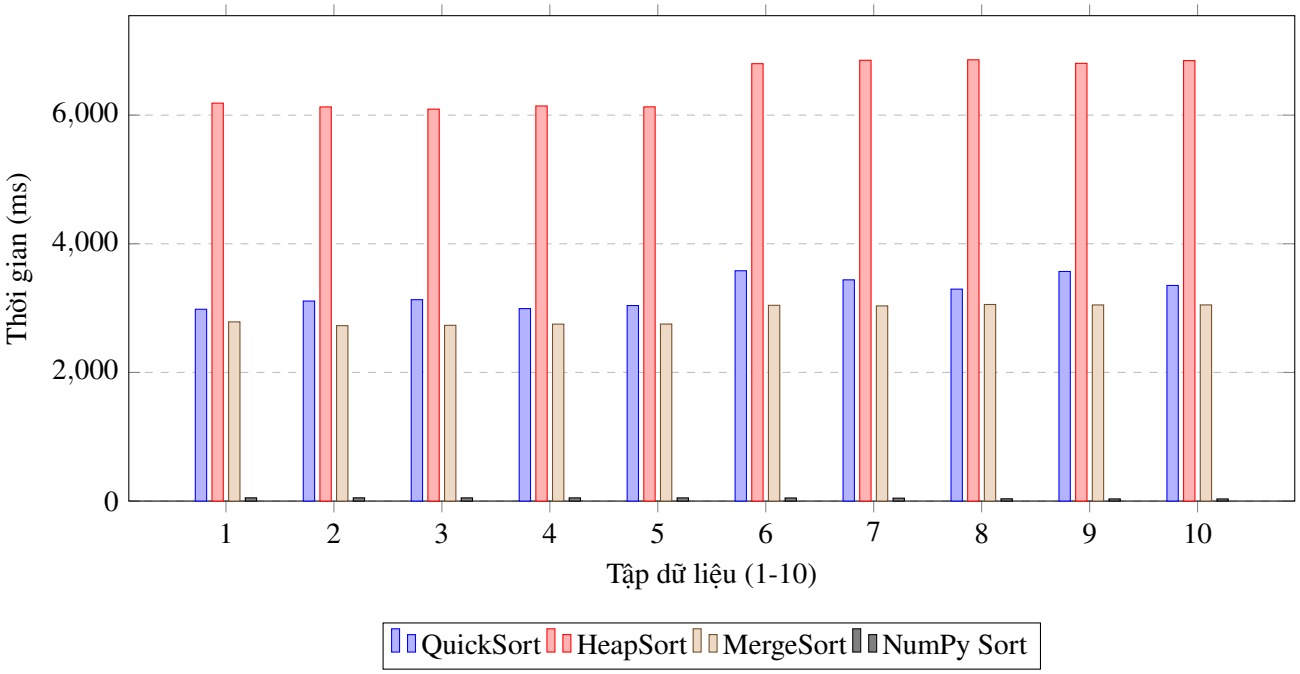
1 Kết quả thử nghiệm

1.1 Bảng thời gian thực hiện

Bảng 1: Bảng thời gian thực hiện các thuật toán sắp xếp (đơn vị: ms)

Dữ liệu	QuickSort	HeapSort	MergeSort	NumPy Sort
1	2982.25	6186.58	2786.01	50.32
2	3109.81	6127.78	2727.07	50.11
3	3131.07	6093.57	2732.45	49.80
4	2992.50	6142.89	2750.80	49.73
5	3039.99	6128.89	2752.36	49.55
6	3580.42	6799.88	3043.71	48.17
7	3439.39	6852.84	3034.07	45.55
8	3295.04	6860.24	3056.20	35.19
9	3569.50	6805.42	3049.18	33.57
10	3352.25	6847.86	3049.64	33.09
Trung bình	**3249.22**	**6484.60**	**2898.15**	**44.51**

1.2 Biểu đồ thời gian thực hiện



Hình 1: Biểu đồ so sánh thời gian thực thi giữa các thuật toán trên 10 tập dữ liệu.

2 Kết luận

Dựa trên kết quả thực nghiệm với bộ dữ liệu gồm 10 dãy số (kích thước 10^6 phần tử/dãy), báo cáo rút ra các nhận định sau:

2.1 Đánh giá hiệu năng tốc độ

Kết quả ghi nhận sự chênh lệch rõ rệt về thời gian thực thi giữa các thuật toán:

- **NumPy Sort ($\approx$ 0.04s):** Đạt hiệu suất cao nhất, vượt trội hoàn toàn so với các thuật toán còn lại nhờ được tối ưu hóa ở tầng thấp.
- **MergeSort ($\approx$ 2.90s):** Trong lần thử nghiệm này, MergeSort cho thấy hiệu năng rất ấn tượng, ổn định và nhanh hơn QuickSort.
- **QuickSort ($\approx$ 3.25s):** Vẫn duy trì tốc độ tốt, tuy nhiên chậm hơn MergeSort một chút trong các trường hợp thử nghiệm này.
- **HeapSort ($\approx$ 6.48s):** Có tốc độ thấp nhất (chậm hơn QuickSort và MergeSort khoảng hơn 2 lần).

2.2 Đánh giá tính ổn định

- Các thuật toán đều duy trì độ phức tạp trung bình trên tập dữ liệu ngẫu nhiên.
- Đối với dữ liệu đặc thù (đã sắp xếp tăng/giảm), việc cài đặt **QuickSort** với kỹ thuật chọn phần tử chốt (Pivot) hợp lý đã giúp thuật toán tránh được trường hợp xấu nhất, duy trì hiệu năng ổn định.

2.3 Tổng kết

Thực nghiệm cho thấy lý thuyết độ phức tạp Big-O chỉ là ước lượng tiệm cận. Trong thực tế, các yếu tố như quản lý bộ nhớ, chi phí ngôn ngữ (Python) và kiến trúc phần cứng ảnh hưởng lớn đến thời gian chạy.

Đối với các bài toán xử lý dữ liệu lớn, việc sử dụng các thư viện chuẩn (như **NumPy**) là giải pháp tối ưu nhất về hiệu năng và độ tin cậy.