Lớp: ATTN2024

**BÁO CÁO KẾT QUẢ THỬ NGHIỆM**

Thời gian thực hiện: 25/02/2025 – 04/03/2025

**Sinh viên thực hiện: Lê Phạm Khánh Linh**

**Nội dung báo cáo:**

1. ***Kết quả thử nghiệm***
   1. ***Bảng thời gian thực hiện[[1]](#footnote-1)***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dữ liệu** | **Thời gian thực hiện (ms)** | | | | |
| **Quicksort** | **Heapsort** | **Mergesort** | **Sort (C++)** | **Sort (numpy)** |
| 1 | 151.855 | 528.51 | 484.41 | 180.495 | 7.461 |
| 2 | 340.759 | 519.796 | 462.186 | 135.607 | 7.604 |
| 3 | 300.636 | 659.367 | 609.779 | 448.947 | 7.435 |
| 4 | 301.072 | 638.242 | 609.966 | 321.87 | 7.387 |
| 5 | 293.367 | 626.767 | 618.016 | 313.602 | 8.452 |
| 6 | 286.841 | 636.915 | 594.103 | 315.631 | 8.197 |
| 7 | 300.326 | 648.987 | 704.701 | 316.327 | 8.582 |
| 8 | 310.015 | 659.246 | 610.559 | 339.084 | 7.492 |
| 9 | 289.846 | 647.541 | 626.449 | 323.582 | 8.108 |
| 10 | 363.871 | 648.373 | 626.537 | 338.83 | 7.579 |
| Trung bình | 293.859 | 621.374 | 594.670 | 303.398 | 7.830 |

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

1. ***Kết luận:***

* Sort của NumPy là thuật toán nhanh nhất với thời gian trung bình 7.830ms, vượt xa các phương pháp khác. Nhờ Timsort kết hợp với vector hóa trong C, NumPy tối ưu hóa hiệu suất với khối lượng lớn.
* Sort của C++ có thời gian trung bình 303.398ms, sử dụng Introsort để điều chỉnh thuật toán theo dữ liệu đầu vào. Điều này giúp C++ sort đạt hiệu suất cao và ổn định, lý tưởng cho các hệ thống xử lý thời gian thực.
* Quicksort có thời gian 293.859ms, nhỉnh hơn một chút so với sort C++. Tuy nhiên, hiệu suất không ổn định do phụ thuộc vào cách chọn pivot, có thể khiến tốc độ dao động đáng kể giữa các bộ dữ liệu.
* Mergesort (594.670ms) và Heapsort (621.374ms) có tốc độ gần như nhau. Mergesort tiêu tốn bộ nhớ phụ để chia nhỏ dữ liệu, trong khi Heapsort phải xử lý cấu trúc heap, làm giảm hiệu suất.
* Nếu cần tốc độ tối đa: NumPy là lựa chọn tốt nhất.
* Nếu cần sự ổn định và đa năng: Sort C++ tối ưu hơn.
* Nếu muốn tùy chỉnh thuật toán sắp xếp: Quicksort là lựa chọn linh hoạt.
* Nếu không ưu tiên hiệu suất: Heapsort và Mergesort vẫn có thể dùng nhưng không phải phương án tối ưu.

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

* Link: https://github.com/lyns184/DSA\_Sort.git

1. Số liệu chỉ mang tính minh họa [↑](#footnote-ref-1)