Bài tập cấu trúc dữ liệu (2022) , 1–4
UIT-VNUHCM

Báo cáo kết quả thực nghiệm

NGƯỜI THỰC HIỆN
Lê Thanh Minh, Trường đại học Công Nghệ Thông Tin ,Đại học quốc gia Hồ Chí Minh, Đường Hàn Thuyên, 71350, Thành phố Hồ Chí Minh, Việt Nam
*Thắc mắc liên hệ :21520063@gm.uit.edu.vn

[]

1. Yêu cầu thực hành

- Tạo bộ dữ liệu gồm 10 dãy, mỗi dãy khoảng 1 triệu số thực (ngẫu nhiên); 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;
- Viết các chương trình sắp xếp dãy theo các thuật toán QuickSort, HeapSort, MergeSort và chương trình gọi hàm sort của C++;[1]
- Chạy thử nghiệm mỗi chương trình đã viết ở trên với bộ dữ liệu đã tạo, ghi nhận thời gian thực thi từng lần thử nghiệm
- Viết báo cáo thử nghiệm: kết quả thử nghiệm ở dạng bảng dữ liệu và dạng biểu đồ; nhận xét kết quả thực nghiệm; báo cáo nộp bằng file PDF
- Toàn bộ các file liên quan cần được lưu trữ trên github (public) và ghi đường dẫn vào trong file báo cáo.[2]

2. Sơ lược về các phép sort

2.1. *Heap sort*

H eap sort là kỹ thuật sắp xếp dựa trên so sánh dựa trên cấu trúc dữ liệu Binary Heap. Nó tương tự như sắp xếp lựa chọn, nơi đầu tiên chúng ta tìm phần tử lớn nhất và đặt phần tử lớn nhất ở cuối. Chúng ta lặp lại quá trình tương tự cho các phần tử còn lại.

2.2. *Quick sort*

Q uick sort là một thuật toán chia để trị (Divide and Conquer algorithm). Nó chọn một phần tử trong mảng làm điểm đánh dấu(pivot). Thuật toán sẽ thực hiện chia mảng thành các mảng con dựa vào pivot đã chọn. Việc lựa chọn pivot ảnh hưởng rất nhiều tới tốc độ sắp xếp (độ phức tạp lên đến $O(n^2)$) và trường hợp tối ưu nhất là chọn pivot ngẫu nhiên.

2.3. *Merge sort*

G iống như Quick sort, Merge sort là một thuật toán chia để trị. Thuật toán này chia mảng cần sắp xếp thành 2 nửa. Tiếp tục lặp lại việc này ở các nửa mảng đã chia. Sau cùng gộp các nửa đó thành mảng đã sắp xếp

---

[1] Std::sort in C++ <algorithm> library
[2] Link Github nộp bài tập ở đây

2.4. *Std::sort*

Std::sort là môt hàm sort trong thư viện chuẩn (Standard library template) của C++, nó có độ phức tạp tương đương các phương pháp sort còn lại

2.5. *Tổng kết*

C ả 4 phép sort này đều có độ phức tạp tối ưu là O(nlog(n)), tuy nhiên đối với quick sort độ phức tạp này lại phụ thuộc vào điểm chọn sắp xếp (pivot), đối với heapsort lại không cần tới đệ quy

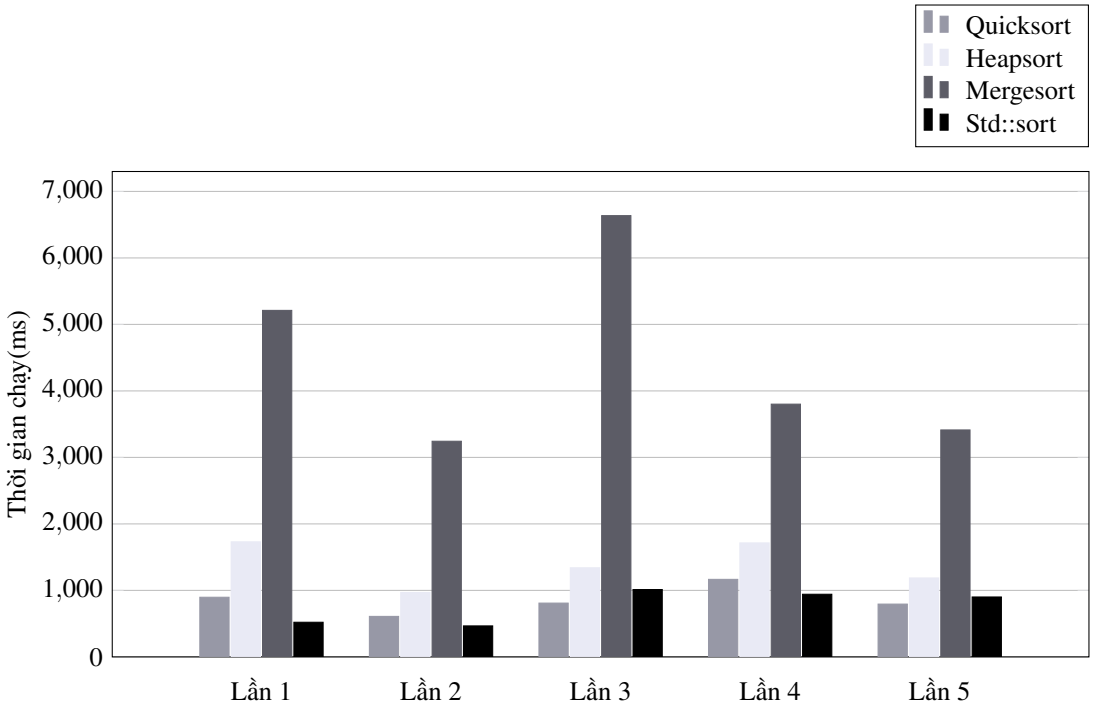
3. Tiến hành đo đạc

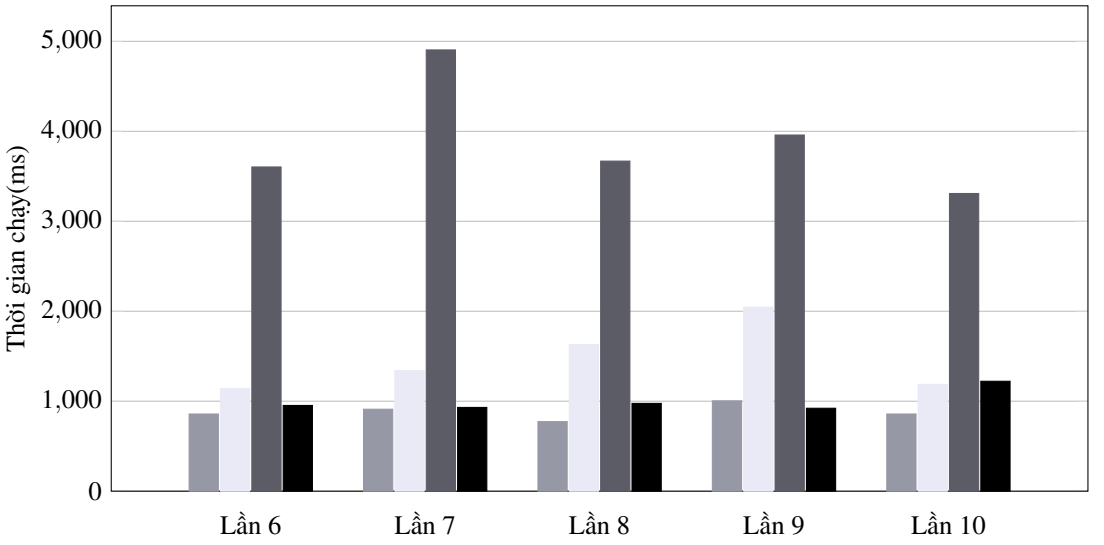
TABLE 1 *Bảng số liệu đo thời gian thực hiện sort*

Thời gian	Lần 1	Lần 2	Lần 3	Lần 4	Lần 5	Lần 6	Lần 7	Lần 8	Lần 9	Lần 10	TB
Quicksort	897	609	809	1166	794	858	911	773	1004	858	867
Heapsort	1730	968	1340	1714	1187	1142	1340	1631	2046	1187	1309
Mergesort	5211	3243	6636	3801	3413	3604	4905	3669	3959	3309	4175
Std::sort	940	690	1702	819	827	865	1121	1154	1163	899	1018

[1] Thực hiện trên chip i3-8130u, IDE visual studio 2019

[2] Thời gian đo khác nhau tùy theo máy và bộ dữ liệu được tạo ngẫu nhiên vào thời điểm đó





4. Tổng kết bài tập - Nhận xét

Sau khi đã tham khảo và tự hoàn thành tất cả các chương, em đã hoàn thành bài tập. Mặc dù hoàn thành nhưng vẫn sẽ có những chỗ sai sót chưa được xử lý,mong thầy cô góp ý thêm

Nếu dữ liệu đã được sắp xếp sẵn thì việc sử dụng std::sort cho thời gian nhanh nhất, còn trong hầu hết trường hợp việc sử dụng quicksort (với pivot hay phần tử được chọn ngẫu nhiên) thì quicksort cho tốc độ nhanh nhất, heapsort chạy lâu hơn 30% so với std::sort và lâu hơn 53% so với quicksort. Mergesort chạy lâu nhất với trung bình là 4,2s

Kết luận : Mergesort cho thời gian chạy lâu nhất, Heapsort cho thời gian chạy trung bình, trong khi đó Quicksort cho tốc độ rất nhanh khi chọn được pivot đẹp còn std::sort có tốc độ chạy tốt

File được tải lên Github

- Link nộp bài : Link Github nộp bài tập ở đây
- *random.cpp*: File chứa các function tạo 1 triệu phần tử double random gồm 10 dòng , 2 dòng đầu đước sắp xếp tăng dần và 8 dòng sắp xếp ngẫu nhiên và xuất ra file *Inputdata.txt* , và một function dùng để đo thời gian của các phép sort
- *Testing.csv* : file ghi lại thời gian thực hiện các thuật toán
- *Inputdata.txt* : Bộ dữ liệu cần sử dụng để testing (Do file >25mb nên không thể tải lên github).
- *SortingAlgorithm.cpp* : file chứa các thuật toán quick sort,heap sort, merge sort đã được hoàn thiện.
- *21520063-report.pdf* : file báo cáo chi tiết kết quả.
- *sort.h* : header chứa các function để chạy
- *main.cpp* : file chạy trên compiler .

Một số nguồn tham khảo

REFERENCES

1. Nguyễn Văn Hiếu, *https://nguyenvanhieu.vn/thuat-toan-sap-xep-merge-sort/*
2. Khanna98, pineconelam, jnjomnsn, mayanktyagi1709, princi singh. *https://www.geeksforgeeks.org/merge-sort/*
3. Heapsort, *https://cafedev.vn/thuat-toan-heapsort-gioi-thieu-chi-tiet-va-code-vi-du-tren-nhieu-ngon-ngu-lap-trinh/*. MIT press, 2012.
4. Rashmi Raj, *Analysis of Algorithm*, Stanford University, 2017
5. Hoare and Lomuto, *Quicksort algorithm's implementation*
6. Max heap visuavisualization *https://visualgo.net/en/heap?slide=1*
7. Đỗ Việt Anh (lionit), *https://vnoi.info/wiki/translate/wcipeg/Binary-Heap.md*