Jaden Liu

Zliu259@ucsc.edu

11/7/2020

CSE13s Fall 2020

Assignment5: Sorting

WRITEUP

Overview:
From this lab, I learned 4 types of sorting algorithms in C. Each method could implement in a unique way of C language, which is different from what I have learnt in Python.

Bubble sort:

Time complexity: Always O(n^2)

Bubble sort is the basic sorting algorithm in programming. It is easy to experiment it in C. But it usually takes longest time, since the most comparisons and moves.

I just mimic the pseudocode provided to do the bubble sort.

Shell sort:

Time complexity: Best O(n), O(n^3/2) when gap is 2^k-1(1,3,7,15…)

Shell sort is a special insertion sort, or it is insertion sort when gap is 1. Since insertion sort is good at almost sorted arrays, so shell sort first makes the array like sorted, then implements the insertion sort. Less moves but more compares to do the shell sort.

I mimic the pseudocode provided most times. However, the C doesn't have *yield* word, so I change the way of writing gap function.

Quick sort:

Time complexity: Average: O(nlogn), Worst: O(n^2)

Quick is a recursive way to do the sorting. It keeps dividing the array to the left, and right, which is smaller and larger than the pivot number. It usually have the least moves and compares when implementing large arrays.

I just mimic the pseudocode provided to do the quick sort.

Binary insertion sort:

Time complexity: Average: O(nlog2n), Worst: O(n^2)

Binary insertion sort improved from insertion sort. It finds the position to insert faster than find each one by one. So, the time saved from that, which from O(n) to O(log2n).

I just mimic the pseudocode provided to do the binary sort.

Conclusion:
I learned a lot from these four sorting methods. Especially I learned the how each sorting algorithm save time on comparisons or moves.