HOMEWORK 4

ALGORITHMS AND DATA STRUCTURES (CH08-320201)

Spring 2018

Prof. Dr. Michael Sedlmair
Computer Science & Electrical Engineering
Jacobs University

Due on Monday, Mar 5, 2018, 23:55.

Problem 1: Bubble Sort & Stable and Adaptive Sorting.s *(3+2+3+2=10 points)*

(a) Bubble Sort is a comparison sorting algorithm that works by repeatedly stepping through the list to be sorted, comparing each pair of adjacent items, and swapping them if they are in the wrong order. This is repeated until no swaps are needed, which indicates that the list is sorted. Write the Bubble Sort algorithm in pseudocode including comments to explain the steps.

(b) Derive the asymptotic worst-case, average-case, and best-case running time of Bubble Sort.

(c) Stable sorting algorithms maintain the relative order of records with equal keys (i.e., values). Thus, a sorting algorithm is stable if whenever there are two records R and S with the same key and with R appearing before S in the original list, R will appear before S in the sorted list. Which of the sorting algorithms Insertion Sort, Merge Sort, Heapsort, and Bubble Sort are stable? Explain your answer.

(d) A sorting algorithm is adaptive, if it takes advantage of existing order in its input. Thus, it benefits from the pre-sortedness in the input sequence and sorts faster. Which of the sorting algorithms Insertion Sort, Merge Sort, Heapsort, and Bubble Sort are adaptive? Explain your answer.

Problem 2: Heap Sort *(8+4+3=15 points)*

(a) *Implement* the Heap Sort algorithm as presented in class.

(b) *Implement* a variant of the Heap Sort that works as follows: In the first step it also builds a max-heap. In the second step, it also proceeds as the Heap Sort does, but instead of calling MAX-HEAPIFY, it always floats the new root all the way down to a leaf level. Then, it checks whether that was actually correct and if not fixes the max-heap by moving the element up again. This strategy makes sense when considering that the element that was swapped to become the new root is typically small and thus would float down to a leaf level in most cases. Hence, one would save the additional tests when floating down the element. And, the fixing step (moving the element upwards again) would be a rare case.

(c) Compare the original Heap Sort and its variant in (b) for input sequences of different lengths (including larger input sequences). What can you observe?

Remarks

Solutions have to be handed in via Moodle by the due date. For late submissions you need to get in contact with the TAs directly. You need to upload one zip-file that contains a PDF-file for the theoretical parts and source files (no executables or object files) for the programming assignments. The source files need to include a makefile. Programming assignments need to be handed in as C, C++, or Python code. **Please write your own code.** It is ok to take snippets from online resources, but they need to be clearly marked.