Homework Assignment 4

Due Date: March 14, 2021, 23:59

Note. Please note that this semester all assignments are group assignments. Further note that for the grading we will apply a "10%" rule, i.e. the maximum number of points for this assignments is 55, but 50 will be counted as 100%. Points that exceed 50 will be stored in a separate counter and used later for compensation of lost points in other assignments or (if not used up this way) the final exam.

Exercise 1.

(i) Show that the running time of `siftUp`(n) is $O(\log n)$ and hence an insert into a heap takes time in $O(\log n)$.

(ii) The `siftDown` used in the *heapsort* algorithm requires about $2 \log n$ comparisons. Show how to reduce this to $\log n + O(\log \log n)$.

Hint: Determine first a path p along which elements need to be swapped, then perform a binary search on this path to find the proper position for the root element.

total points: 11

Exercise 2. Let R be a binary relation on the set of integers.

(i) Use an associative array to represent R such that it becomes easy to check whether R is symmetric.

(ii) Implement your symmetry checking algorithm.

total points: 11

Exercise 3. *Pairing heaps* are an efficient data structure using a very simple technique for rebalancing, though a full theoretical analysis is missing. They rebalance only in connection with the *deleteMin* operation. If $r_1, \ldots, r_k$ is the sequence of root nodes stored in a location `roots`, then *deleteMin* combines r_1 with r_2, r_3 with r_4, etc., i.e. the roots are paired.

(i) Explain how to implement pairing heaps using three pointers per heap item i: one to the oldest child (i.e. the child linked first to i), one to the next younger sibling (if any), and one to the next older sibling. If there is no older sibling, the third pointer goes to the parent.

(ii) Explain how to implement pairing heaps using only two pointers per heap item: one to the oldest child and one to next younger sibling. If there is no younger sibling, the second pointer goes to the parent.

Figure 6.8 in the [Mehlhorn 2008] textbook (available on BB) illustrates pairing heaps with three or two pointers per heap item.

total points: 17

Exercise 4. A *McGee heap* has the same structure as a Fibonacci heap, but supports just the mergeable heap operations. The implementations of the operations are the same as for Fibonacci heaps, except that insertion and union consolidate the root list as their last step. What are the worst-case running times of operations on McGee heaps?

total points: 16