University of Zurich
Department of Informatics
Database Technology Group
Prof. Dr. Michael Böhlen

Informatics II
Exercise 5

March 16, 2022

Heap and Heapsort

Task 1

Task 1.1. Given the input array A = [46 77 55 38 41 85], what is the state of A after the algorithm `BuildHeap(A, 6)` that builds a max-heap is excuted?

- A.[85 77 55 41 38 46]
- B.[38 41 46 77 55 85]
- C.[85 55 77 38 41 46]
- D.[85 77 55 38 41 46]

Task 1.2. A heap has array representation and tree representation (nearly complete binary tree). Run the `HeapSort` algorithm on the array A = [11 0 9 19 8 1 5 13 18 7] to sort elements of A in an ascending order. Show states of tree representation and array representation when the state of A changes.

Task 1.3. Implement `Heapify(A,i,s)`, `BuildHeap(A,n)` and `HeapSort(A,n)` in C. Use the array A in task 1.2 as the input array to print A's array representation.

Task 1.4. How many times the function `Heapify` has been executed in HeapSort to completely sort an array?

- A.n/2
- B.(3n-2)/2
- C.n-1
- D.n

Task 1.5. What's the worst case time complexity of HeapSort?

- A.O(n)
- B.O($2n$)
- C.O($n \log n$)
- D.O(n^2)

Quicksort

Task 2.1. Run quicksort algorithms using Lomuto partition and Hoare partition on the input array A = [11 0 9 19 8 1 5 13 18 7]. Show state of A when A changes.

Task 2.2. Implement LomutoPartition(A,l,r), HoarePartition(A,l,r) and QuickSort(A,l,r) taught in class in C code. Using the array A of task 2.1 as the input to check the correctness of your implementation.

Task 2.3. In which case, heap short is faster than quick sort?

Task 3

Partitioning plays a crucial role in Quicksort algorithm. What are the issues of always choosing the middle element at the position $n/2$ of an array of n elements as the pivot?

Tasks in 2020 FS Midterm 1

The values **3 2 ; 9 ; 10 8 7 1 ;** are inserted in the given order into the same `max-heap`. The `max-heap` is initially empty. Draw the `max-heap` at the positions marked by a semicolon.

The key element of the quicksort algorithm is its partitioning procedure. Complete the boxes below to complete algorithm Partition(A, l, r) that partitions array $A[l..r]$.

Algorithm: Partition(A,l,r)

x = A[l];
i = l;

for j = ☐ **to** ☐ **do**

  if *A[j]* ☐ x **then**

    i = ☐ ;

    exchange A[i] and A[j];

exchange ☐ and ☐ ;

return ☐ ;