Exercise Sheet 5
COMS10007 Algorithms 2019/2020

21.04.2020

1 Heap Sort

Consider the following array A:

4	3	9	10	14	8	7	2	1	7

1. Interpret A as a binary tree as in the lecture (on heaps).
2. Run Create-Heap() on the initial array. Give the sequence of node exchanges. Draw the resulting heap.
3. What is the worst-case runtime of Heapify()?
4. Explain how heap sort uses the heap for sorting. Explain why the algorithm has a worst-case runtime of $O(n \log n)$.
5. Give an array of length n so that heap-sort runs in $O(n)$ time on A.

2 Merge Sort

Illustrate how the Mergesort algorithm sorts the following array using a recursion tree:

$$11 \quad 7 \quad 2 \quad 5 \quad 9 \quad 6 \quad 1$$

3 Quick Sort

Consider an array A of length n so that $A[i] = n - i$. For example, for $n = 10$ we are given the following array:

$$A = 10 \quad 9 \quad 8 \quad 7 \quad 6 \quad 5 \quad 4 \quad 3 \quad 2 \quad 1 \; .$$

The goal is to sort A in non-decreasing order which in this case is equivalent to reversing it. The pivot plays a central role in Quicksort. Consider the following options as a choice for the pivot:

1. The right-most position.
2. The element at position $\lceil n/2 \rceil$.
3. The left-most position.

For each of these options, what is the runtime of Quicksort on A? State your answers using $\Theta(.)$-notation. Justify your answers.

4 Countingsort and Radixsort

1. Illustrate how Countingsort sorts the following array:

4	2	2	0	1	4	2

2. Illustrate how Radixsort sorts the following binary numbers:

   100110 101010 001010 010111 100000 000101

3. Radixsort sorts an array A of length n consisting of d-digit numbers where each digit is from the set $\{0, 1, \ldots, b\}$ in time $O(d(n + b))$.

   We are given an array A of n integers where each integer is *polynomially bounded*, i.e., each integer is from the range $\{0, 1, \ldots, n^c\}$, for some constant c. Argue that Radixsort can be used to sort A in time $O(n)$.

   Hint: Find a suitable representation of the numbers in $\{0, 1, \ldots, n^c\}$ as d-digit numbers where each digit comes from a set $\{0, 1, \ldots, b\}$ so that Radixsort runs in time $O(n)$. How do you chose d and b?

5 Loop Invariant for Radixsort

Radixsort is defined as follows:

Require: Array A of length n consisting of d-digit numbers where each digit is taken from the set $\{0, 1, \ldots, b\}$
1: **for** $i = 1, \ldots, d$ **do**
2: Use a stable sort algorithm to sort array A on digit i
3: **end for**

(least significant digit is digit 1)

In this exercise we prove correctness of Radixsort via the following loop invariant:

At the beginning of iteration i of the for-loop, i.e., after i has been updated in Line 1 but Line 2 has not yet been executed, the following holds:

The integers in A are sorted with respect to their last $i - 1$ digits.

1. *Initialization:* Argue that the loop-invariant holds for $i = 1$.

2. *Maintenance:* Suppose that the loop-invariant is true for some i. Show that it then also holds for $i + 1$.

   Hint: You need to use the fact that the employed sorting algorithm as a subroutine is stable.

3. *Termination:* Use the loop-invariant to conclude that A is sorted after the execution of the algorithm.