DESIGN AND ANALYSIS OF ALGORITHMS — HT 2019

Problem Sheet 2

Questions marked with $*$ are not intended to be discussed in tutorials, answers to these questions will be posted on the course webpage.

Divide and Conquer, cont'd

Question 1

Assume a linear-time procedure PARTITION'(A, p, r, x) that takes an array $A[p\,..\,r)$ of distinct integers and an array element x as input. The procedure partitions the array around x, returning an index q with $p \leq q < r$ such that $A[q] = x$, $A[p\,..\,q)$ consists of elements less than x, and $A[q+1\,..\,r)$ consists of elements greater than x.
Assume also that you have a "black box" worst-case linear-time median subroutine MEDIAN'(A, p, r) that takes an array $A[p\,..\,r)$ of distinct integers and returns the $\lceil (r-p)/2 \rceil$-order statistic of $A[p\,..\,r)$.

(a) Using PARTITION' and MEDIAN' give the pseudocode of a quicksort-like sorting algorithm that runs in $O(n \log n)$ time in the worst case, assuming that all elements are distinct. Justify your answer.

(b) Using PARTITION' and MEDIAN' give the pseudocode of a linear-time algorithm that solves the selection problem (for an arbitrary order statistic). Justify your answer.

(c) Show how you could use one call to MEDIAN' to solve the selection problem (for an arbitrary order statistic) by using some "padding". Justify your answer.

Question 2

$*$ For any even integer $n > 0$ it is always possible to find integers m and k such that m is odd and $n = m \cdot 2^k$. For such an n, the product of two $n \times n$ matrices can be computed in the following way. We use Strassen's method recursively down to $m \times m$ matrices and at that stage we switch to the conventional method rather than continuing with Strassen's method right down to 1×1 matrices. Compare this hybrid method with the conventional method for $n = 800$.

Question 3

Suppose that two $n \times n$ matrices can be multiplied by performing 32 block multiplications and 144 block additions of $n/4 \times n/4$ matrices. For simplicity you may assume that $n = 4^k$.

(a) Determine the asymptotic complexity of the recursive algorithm based on this fact, giving an upper bound for the constant factor. (Count additions and multiplications.)

(b) Estimate how large n has to be to make this algorithm preferable to the conventional algorithm.

Heaps, heapsort and priority queues

Question 4

(a) What are the minimum and maximum numbers of elements in a heap of height h?

(b) Where in a max-heap might the smallest element reside, assuming that all elements are distinct?

(c) Is an array that is in sorted order a min-heap?

(d) Is the sequence $[\,23, 17, 14, 6, 13, 10, 1, 5, 7, 12\,]$ a max-heap?

Question 5

Show that the worst-case running time of MAX-HEAPIFY on a heap of size n is $\Omega(\log n)$.
(*Hint*. For a heap with n nodes, give node values that cause MAX-HEAPIFY to be called recursively at every node on a path from the root down to a leaf.)

Question 6

Give an algorithm for removing an *arbitrary* element from a heap of size n. Describe your algorithm in pseudocode or in English and determine its worst-case time complexity.

Question 7

$*$ What is the running time of Heapsort on an array of length n that is already sorted in increasing order? What about decreasing order?

Question 8

Give an $O(n \log k)$-time algorithm to merge k sorted lists into one sorted list, where n is the total number of elements in all the input lists. (*Hint:* Use a heap for k-way merging.)