DESIGN AND ANALYSIS OF ALGORITHMS — HT 2019

Problem Sheet 1

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

Big-O and other asymptotic notations

Question 1

Let $a(n) = 10^6 n^2$ and $b(n) = 10^n$. Computer A performs 10^6 operations per second; computer B performs 10^{12} operations per second. In the worst case on an instance of size n, an implementation of an algorithm α solves a problem P in $a(n)$ operations on computer A, and an implementation of an algorithm β solves P in $b(n)$ operations on computer B.

(a) Which instances of P would you solve using the implementation of α on A, and which using the implementation of β on B?

(b) Estimate how long it would take in the worst case to solve an instance of P of size 30 using α on A and using β on B.

Question 2

$*$ Suppose that k is a positive integer. Show that if $f = O(n^k)$ then there are constants $a, b > 0$ such that $f(n) \leq an^k + b$ for all $n \geq 0$.

Question 3

Give yes/no answers to the following:

	$f(n)$	$g(n)$	$f = O(g)$?	$f = \Omega(g)$?	$f = \Theta(g)$?
$a.$	$n - 100$	$n - 200$			
$b.$	$n^{1/2}$	$n^{2/3}$			
$c.$	$100n + \log n$	$n + (\log n)^2$			
$d.$	$n \log n$	$10n \log 10n$			
$e.$	$\log 2n$	$\log 3n$			
$f.$	$n^{0.1}$	$(\log n)^{10}$			
$g.$	$\sqrt{n}$	$(\log n)^3$			
$h.$	$n2^n$	3^n			
$i.$	2^n	2^{n+1}			
$j.$	$(\log n)^{\log n}$	$2^{(\log n)^2}$			

Question 4

Show that $\log(n!) = \Theta(n \log n)$.

Recurrences

Question 5

(a) $*$ Suppose that $f_0 = O(1)$ and that for $k > 0$ and $n > 0$

$$f_k(n) \ \leq \ f_k(n-1) + f_{k-1}(n).$$

Show that $f_k = O(n^k)$ for $k \geq 0$.

(b) $*$ Suppose that $g_0 = \Omega(1)$ and that for $k > 0$ and $n > 0$

$$g_k(n) \ \geq \ g_k(n-1) + g_{k-1}(n).$$

Show that $g_k = \Omega(n^k)$ for $k \geq 0$.

Question 6

Solve the following recurrences, given $T(1) = 1$, to obtain asymptotic upper bounds on $T(n)$:

(a) $T(n) \leq 2T(n-1) + n$

(b) $T(n) \leq T(n/2) + n \log n$

(c) $T(n) \leq T(n-1) + 3n^2$

(d) $T(n) \leq 2T(n/2) + n^2$

Comparison problems: Searching, sorting, selection

Question 7

(a) Show how to find the largest and the smallest among four integers using four comparisons between integers, that is, four comparisons each of which involves just two integers.

(b) Hence design a divide-and-conquer algorithm that finds the largest and the smallest among n integers using at most $3n/2 - 2$ comparisons between integers, where $n \geq 2$ is a power of 2. Justify your answer using induction on $k \geq 1$ where $n = 2^k$.

Question 8

A "ternary" search algorithm tests the element at position $n/3$ for equality with some value x and then possibly checks the element at $2n/3$ either discovering x or reducing the set size to one third of the original. Compare this with binary search.

Question 9

Given two sorted lists (stored in arrays) of size n, find an $O(\log n)$ algorithm that computes the n-th largest element in the union of the two lists.

Question 10

$*$ Let $X = \langle x_0, x_1, \cdots, x_{n-1} \rangle$ be a *cyclically sorted* sequence of integers, i.e. one where

$$\exists\, 0 \leq j < n \,.\, \forall\, 0 \leq i < n-1 \,.\, x_{(j+i) \bmod n} < x_{(j+i+1) \bmod n}$$

Show that $O(\log n)$ binary comparisons are sufficient to determine whether the sequence X contains the integer z.

Question 11

Describe a $\Theta(n \log n)$-time algorithm that, given n integers stored in an array $A[1 \mathinner{..} n]$ and another integer z, determines whether or not there exist $1 \leq i, j \leq n$ such that $A[i] + A[j] = z$.

Question 12

Let $A[1 \mathinner{..} n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$ then the pair (i, j) is called an *inversion* of A. Give an algorithm that determines the number of inversions in any permutation on n elements in $\Theta(n \log n)$ worst-case time. (*Hint.* Modify merge sort.)
What, if anything, needs to be changed if A may contain duplicates?