Albert-Ludwigs-Universität, Inst. für Informatik
Prof. Dr. Fabian Kuhn
H. Ghodselahi, O. Saukh October 21, 2015

Algorithm Theory, Winter Term 2015/16
Problem Set 1

hand in (hard copy or electronically) by 10:15, Thursday October 29, 2015, tutorial session will be on November 2, 2015

Exercise 1: Complexity (2+2 points)

Characterize the relationship between $f(n)$ and $g(n)$ in the following two examples by using the O, Θ, or Ω notation. Hence, state if $f(n) = \Theta(g(n))$ and otherwise if $f(n) = O(g(n))$ or $f(n) = \Omega(g(n))$. Explain your answers (formally)!

a) $f(n) = n^{\epsilon}$ (for any positive constant $\epsilon < \frac{1}{2}$) $g(n) = \log n$

b) $f(n) = \log n!$ $g(n) = n \log n$

Exercise 2: Recurrence Relations (2+2 points)

a) Guess the solution of the following recurrence relation by repeated substitution.

$$T(n) \leq 2 \cdot T(\frac{n}{2}) + c \cdot n \log_2 n, \quad T(1) \leq c$$

where $c > 0$ is a constant.

b) Use induction to show that your guess is correct.

Exercise 3: Maximum Sum Subsequence (4 points)

Given an integer array $A = \{x_0, x_1, \ldots, x_{n-1}\}$ where $x_i \in \mathbb{Z}$ for all $i \in \{0, 1, \ldots, n-1\}$ (note that the values x_i can be negative).

Devise an efficient **divide-and-conquer** algorithm to find a contiguous subsequence of elements in A with maximum possible sum. That is, you need to find indices $0 \leq i_1 \leq i_2 \leq n-1$ such that the sum $\sum_{i=i_1}^{i_2} x_i$ is maximized. Write down the recurrence relation which describes the running time of your algorithm and use the Master theorem to derive the running time of your algorithm.

Hint: *There is a divide-and-conquer solution which runs in time $O(n)$, an $O(n \log n)$ solution gives partial points. In order to find an $O(n)$ algorithm, it might help to first design an $O(n \log n)$ algorithm.*