University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
S. Faour, M. Fuchs, A. Mályusz

Algorithm Theory
Exercise Sheet 1

Due: Friday, 25th of October, 2024, 10:00 am

Exercise 1: Recurrence Relation *(4 Points)*

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

$$T(n) \leq 3 \cdot T\left(\frac{n}{3}\right) + c \cdot n \log_3 n, \quad T(1) \leq c$$

where $c > 0$ is a constant. *(2 Points)*

b) Use induction to show that your guess is correct. *(2 Points)*

Remark: You can assume that n equals 3^j for some $j \in \mathbb{N}$.

Exercise 2: Fraud Detection *(6 Points)*

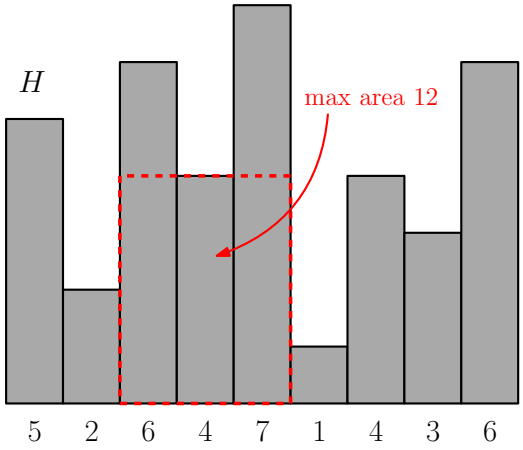
Suppose you're consulting for a bank that's concerned about fraud detection, and they come to you with the following problem. They have a collection of n bank cards that they've confiscated, suspecting them of being used in fraud. Each bank card is a small plastic object, containing a magnetic stripe with some encrypted data, and it corresponds to a unique account in the bank. Each account can have many bank cards corresponding to it, and we'll say that two bank cards are equivalent if they correspond to the same account. It's very difficult to read the account number off a bank card directly, but the bank has a high-tech "equivalence tester" that takes two bank cards and, after performing some computations, determines whether they are equivalent.

Devise an algorithm that answers the following question: *Let $0 < k \leq n$ be an integer, among the collection of n cards, is there a set of more than n/k of them that are all equivalent to one another and if so how many are they?* Your algorithm should use only $O(kn \log n)$ invocations of the equivalence tester. Argue correctness and running time. *(6 Points)*
Remark: Assume that the only feasible operations you can do with the cards are to pick two of them and plug them in to the equivalence tester.

Bonus Question: Let $k = 2$. What happens in this case in part (a)? Can you speed things up by giving a linear time algorithm that solves the problem for this case? If yes, then argue correctness, and analyze its running time. *(You might get extra Points)*
Hint: try reducing the size of the array to at most half via pairing up elements.

Exercise 3: Maximum Rectangle in a Histogram *(10 Points)*

Consider a sequence $h_1, \ldots, h_n$ of positive, integer numbers. This sequence represents a histogram H consisting of n horizontally aligned bars each of width 1, where h_i represents the height of the i^{th} bar. The goal is to find a rectangle of maximum area completely within H (i.e., within the union of bars).



(a) Describe an algorithm that computes a maximum area rectangle in H in time $\mathcal{O}(n^2)$. *(3 Points)*

(b) Describe an algorithm that computes a maximum area rectangle that is within H and also intersects the i^{th} bar in time $\mathcal{O}(n)$ and prove the running time. *(5 Points)*

Remark: Correct solutions in $o(n^2)$ grant partial points.

(c) Give an algorithm that uses the divide and conquer principle to compute a maximum area rectangle in H in time $\mathcal{O}(n \log n)$ and prove the running time. *(2 Points)*

Remark: You can use part (b), even if you did not succeed.