Technische Universität München
Fakultät für Informatik
Lehrstuhl für Algorithmen und Komplexität
Prof. Dr. Harald Räcke
Richard Stotz

Winter term 2018/19
Assignment 5
November 12, 2017

Efficient Algorithms and Data Structures I

*Deadline: November 19, 10:15 am in the **Efficient Algorithms** mailbox.*

Homework 1 (6 Points)

Solve the following recurrence relations using generating functions:

(a) $a_n = -2a_{n-1} - a_{n-2}$ for $n \geq 2$ with $a_0 = 1$ and $a_1 = -1$.

(b) $a_n = a_{n-1} + 2^{n-1} + 3^n$ for $n \geq 1$ with $a_0 = 0$.

Homework 2 (5 Points)

Let $f_0 = 0$ and $f_n = 1/n$ for $n > 0$. The harmonic number h_n is defined as $\sum_{i=0}^{n} f_i$. Use the tables on generating functions (p. 102 and 103) to determine the following.

(a) Give a closed-form expression for $F(z) = \sum_{n\geq 0} f_n z^n$.

(b) Use (a) to determine $\sum_{n\geq 0} h_n/4^n$.

Homework 3 (3 Points)

Give tight asymptotic bounds for the following recurrence relation:

$$T(n) = T(\sqrt{n}) + 1$$

Homework 4 (6 Points)

For some reason, we want to determine tight asymptotic upper and lower bounds for

$$T(n) = T\left(\frac{n}{\log n}\right) + 1 \quad .$$

We first consider the auxiliary recurrence

$$H(m) = H(m - \log m) + 1 \quad .$$

(a) Show by induction that $H(m) \in \Omega\left(\frac{m}{\log m}\right)$.

(b) Show by induction that $H(m) \in \mathcal{O}\left(\frac{m}{\log m}\right)$.

Hint: You may use without proof the fact that $\log(m - \log m) \geq \log m - \frac{(\log m)^2}{2m}$ for $m \geq 256$.

(c) Use the results on $H(m)$ to give tight asymptotic upper and lower bounds for $T(n)$.

Tutorial Exercise 1

(a) Solve the recurrence

$$g_0 = 1;$$
$$g_n = \sum_{i=1}^{n} i \cdot g_{n-i} \text{for } n \geq 1.$$

Hint: Use fact that

$$\sum_{n \geq 0} F_{2n} z^n = \frac{z}{1 - 3z + z^2} \quad ,$$

where F_n is the nth Fibonacci number.

(Extra) Prove the hint!

Tutorial Exercise 2

The *depth* of a node v in a binary search tree is the number of edges on the shortest path from v to the root of the tree.
Show that there exists a binary search tree with n nodes with height in $\omega(\log(n))$ and average depth in $\mathcal{O}(\log(n))$.

Tutorial Exercise 3

In this exercise, we show that the *rotation distance* between binary trees of n nodes is $\mathcal{O}(n)$. For trees T_1 and T_2 over the same nodes, the rotation distance is defined as the number of rotations needed to transform tree T_1 into tree T_2.

(a) A *right-linear chain* is a tree in which every internal node including the root has no left child. Show that any binary tree of n nodes can be transformed into a right-linear chain using at most n rotations.

(b) Conclude that the rotation distance is only $\mathcal{O}(n)$.

```
A generating function is a device somewhat
similar to a bag.  Instead of carrying many
little objects detachedly, which could be
embarrassing, we put them all in a bag, and
then we have only one object to carry, the
bag.
                                  - G. Pòlya
```