Technische Universität München
Fakultät für Informatik
Lehrstuhl für Theoretische Informatik
Prof. Dr. Harald Räcke
Richard Stotz, Dennis Kraft

Winter term 2016 / 2017
Assignment 2
October 31, 2016

Efficient Algorithms and Data Structures I

*Deadline: November 7, 10:15 am in the **Efficient Algorithms** mailbox.*

Homework 1 (6 Points)

Give tight asymptotic upper and lower bounds for $T(n)$, where $T(0)$ is an arbitrary constant, for the following recurrence relations

1. $T(n) = T(n-1) + n^2 \qquad \text{for } n \geq 1$
2. $T(n) = T(n/2) + T(n/4) + T(n/8) + n\ . \qquad \text{for } n \geq 1$

As argued in the lecture you may ignore the fact that function arguments can be non-integer.

Homework 2 (4 Points)

Give tight asymptotic upper and lower bounds for $T(n)$ with

1. $T(n) = 2T(n/4) + \sqrt{n}$
2. $T(n) = 7T(n/3) + n^2$

Homework 3 (5 Points)

Given two $n \times n$ matrices A and B where n is a power of 2, we know how to find $C = A \cdot B$ by performing n^3 multiplications. Now let us consider the following approach. We partition A, B and C into equally sized block matrices as follows:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

Consider the following matrices:

$$\begin{aligned} M_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\ M_2 &= (A_{21} + A_{22})B_{11} \\ M_3 &= A_{11}(B_{12} - B_{22}) \\ M_4 &= A_{22}(B_{21} - B_{11}) \\ M_5 &= (A_{11} + A_{12})B_{22} \\ M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ M_7 &= (A_{12} - A_{22})(B_{21} + B_{22}) \end{aligned}$$

Then,

$$C_{11} = M_1 + M_4 - M_5 + M_7$$

1. Construct the matrices C_{12}, C_{21} and C_{22} from the matrices M_i, as demonstrated for C_{11}.

2. Design an efficient algorithm for multiplying two $n \times n$ matrices based on these facts. Analyze its running time.

Homework 4 (5 Points)

Late in autumn, the squirrel Alexander wants to sort all nuts that he collected over the summer by their size. He uses the traditional algorithm SQUIRREL-SORT (Algorithm 1), which is as follows

Algorithm 1: SQUIRREL-SORT(A, i, j)

if $(A[i] > A[j])$ **then**
    swap $A[i] \leftrightarrow A[j]$
if $i + 1 \geq j$ **then**
    return
$k \leftarrow \lfloor (j - i + 1)/3 \rfloor$
SQUIRREL-SORT$(A, i, j - k)$
SQUIRREL-SORT$(A, i + k, j)$
SQUIRREL-SORT$(A, i, j - k)$

1. Argue that *SQUIRREL-SORT*$(A, 1, n)$ correctly sorts a given array $A[1 \dots n]$. Use induction over the array length.

2. Analyze how much time Alexander asymptotically needs to sort his n nuts using a recurrence relation.

Tutorial Exercise 1

Solve the following recurrence relation without using generating functions:

$$a_n = a_{n-1} + 3^n \qquad \text{for } n \geq 1 \text{ with } a_0 = 6 \;.$$

```
[The master theorem] is ap-
propriate both as a class-
room technique and as tool
for practicing algorithm
designers.
      - Bentley, Haken, Saxe
```