HOMEWORK 3

ALGORITHMS AND DATA STRUCTURES (CH08-320201)

Spring 2018

Prof. Dr. Michael Sedlmair
Computer Science & Electrical Engineering
Jacobs University

Due on Monday, Feb 26, 2018, 23:55.

Problem 1: Fibonacci numbers *(6+4+2+3=15 points)*

(a) Implement all four methods to compute Fibonacci numbers that were discussed in class: (1) naive recursive, (2) bottom up, (3) closed form, and (4) using a matrix representation.

(b) Sample and measure the running times of all four methods for increasing n. For each method, stop the sampling when the running time exceeds 5 seconds. Create a table with your results (max. 1 page).

Tip: the "space" between samples should increase the larger n gets,
e.g. $n \in \{1, 2, 3, 4, 5, 6, 8, 10, 13, 16, 20, 25, 32, 40, 50, 63, ...\}$.

(c) For the same n, do all methods always return the same Fibonacci number? Justify your answer.

(d) Plot your results in a line plot, so that the four approaches can be easily compared. Briefly interpret your results.

Tip: Use log scales for your plot.

Problem 2: Divide & Conquer and Solving Recurrences *(2+4+1+3+1=11 points)*

Consider the problem of multiplying large integers a and b with n bits each. You can assume that addition, substraction, and bit shifting can be done in linear time, i.e., in $\Theta(n)$.

(a) Derive the asymptotic time complexity in the number of bits n for a brute-force implementation of the multiplication.

(b) Derive a divide & conquer algorithm for the given problem by splitting the problem into two subproblems. For simplicity you can assume n to be a power of 2.

(c) Derive a recurrence for the time complexity of the divide & conquer algorithm in (b).

(d) Solve the recurrence in (c) using the recursion tree method.

(e) Validate the solution in (d) by using the master theorem to solve the recurrence.

Remarks

Solutions have to be handed in via Moodle by the due date. For late submissions you need to get in contact with the TAs directly. You need to upload one zip-file that contains a PDF-file for the theoretical parts and source files (no executables or object files) for the programming assignments. The source files need to include a makefile. Programming assignments need to be handed in as C, C++, or Python code. **Please write your own code.** It is ok to take snippets from online resources, but they need to be clearly marked.