Algorithms and Data Structures
Jacobs University Bremen
Dr. Florian Rabe

Homework 11
given: 2017-05-02

Homework 11

You have to submit your solutions as announced in the lecture.
Unless mentioned otherwise, all problems are due 2017-05-11, 11:00.
There will be no deadline extensions unless mentioned otherwise in the lecture.

Problem 11.1 *Tail Recursion* Points: 6

Give a tail-recursive definition of the function $map[A](x : List[A], f : A \to B) : List[B]$ of lists.

The following partial solution may help:

fun $map[A](x : List[A],\ f : A \to B)\ =$
$\quad mapAux(x, f, \qquad\quad)$

fun $mapAux[A](x : List[A],\ f : A \to B,\ result : List[B])\ =$

Solution:

fun $map[A](x : List[A],\ f : A \to B)\ =$
$\quad mapAux(x, f, Nil)$

fun $mapAux[A](x : List[A],\ f : A \to B,\ result : List[B])\ =$
$\quad$ **match** x
$\qquad Nil \mapsto reverse(result)$
$\qquad Cons(hd, tl) \mapsto mapAux(tl, f, Cons(f(hd), result))$

The additional argument $result$ accumulates the argument, and the base case returns it.

To be linear instead of quadratic, the accumulator collects the result in reverse order, and the base case undoes the reversal.

Problem 11.2 *Backtracking* Points: 8

Write a program that finds a solution to the n-queens problems (on an $n \times n$ board) using the general backtracking algorithm.

Problem 11.3 *Divide and Conquer* Points: 8

Implement Karatsuba's divide-and-conquer algorithm for the multiplication of two polynomials of degree $2^n - 1$ as described in the notes.

Problem 11.4 *Master Theorem* Points: 6

Apply the master theorem to derive the Θ-class of the time complexity of

1. mergesort
2. binary search
3. Karatsuba multiplication of polynomials

Show your work.

Solution: We use d, r, and c as in the statement of the Master theorem.

1. $d = 2$ and $r = 2$. We know dividing takes constant and merging liner time, so $c = 1$. Then $r = d^c$, and we obtain $\Theta(n \log_2 n)$.

2. $d = 2$ and $r = 1$. We know dividing takes constant time and no merging is necessary, so $c = 0$. Then $r = d^c$, and we obtain $\Theta(\log_2 n)$.
3. $d = 2$ and $r = 3$. We know dividing takes linear time (splitting and adding polynomials) and merging linear time (adding and shifting polynomials), so $c = 1$. Then $r > d^c$, and we obtain $\Theta(n^{\log_d r}) = \Theta(n^{\log_2 3}) \approx \Theta(n^{1.58})$.