Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. S. Albers
S. Lauer

November 5, 2007

Algorithms Theory, Winter Term 07/08
Assignment 2

hand in by Monday, November 19, 2007, 14 p.m.
(boxes in building 051)

Exercise 1: FFT (5 points)
Compute the product of the two polynomials

$$p(x) = 5x + 2 \quad \text{and} \quad q(x) = 6x + 1$$

using the Fast Fourier Transform. Specify all (recursive) calls of the `FFT` algorithm as well as the ouputs and the assignments of the temporary variables used during the execution.

Exercise 2: FFT (5 points)
Let A and B be two sets of integers in the range $[0, m-1]$ where m is a power of two. Show that the following can be computed in $O(m \log m)$ time with a single `DFT`:

(i) all elements contained in the set $A + B = \{a + b \mid a \in A \wedge b \in B\}$

(ii) for each element $c \in A + B$ the number $k_c = |\{(a, b) \in A \times B : a + b = c\}|$.

Hint: Find some polynomials p_A, p_B of degree less than m that represent the sets A and B.

Exercise 3: Randomized Quicksort (5 points)

a) Let T be the representation of a certain execution of *Randomized Quicksort* as a tree. Describe the relation between the element of a node and the elements of its left and right child. Give a short reason for your answer.

b) Does every permutation π that can arise for an execution of *Randomized Quicksort* on n elements ($n > 2$) occur with probability $\frac{1}{n!}$? Give reasons for your answer.
Hint: For a better understanding of the construction of a permutation for a tree T have a closer look at the example in the lecture.

Exercise 4: RSA (5 points)
For an RSA encryption choose $p = 31$ and $q = 17$. Moreover, let $e = 131$.

a) Compute the number d and specify the outputs of the algorithm `extended-Euclid`. Furthermore, give the public and the private key.

b) Generate a digital signature for the message $M = 72$. What does a recipient of the message have to check in order to verify the signature?
Hint: For generating the signature, use the fast exponentiation algorithm `power` but omit the check for square roots of 1 (modulo n).