Prof. Friedrich Eisenbrand
Alfonso Cevallos and Yuri Faenza

Discussions from: March 04, 2014

Computer Algebra

Spring 2014

Assignment Sheet 2

Exercises marked with a $\star$ can be handed in for bonus points. Due date is March 18.

Exercise 1
Let $f : \mathbb{N} \to \mathbb{R}_+$ be a function with $f(a) + f(b) \leq f(a+b)$. Show that $f(1) + f(2) + f(4) + f(8) + \ldots + f(n) = O(f(n))$.

Exercise 2
Let $x \in \mathbb{R}$ and $n \in \mathbb{N}_{\geq 1}$. Show that $\lfloor \lfloor x \rfloor / n \rfloor = \lfloor x/n \rfloor$; in particular, $\lfloor \lfloor a/b \rfloor / c \rfloor = \lfloor a/bc \rfloor$ for all positive integers a, b, c.

Exercise 3
Let $a, b \in \mathbb{N}$ be odd numbers with $a - b = 2^k$ for some $k \in \mathbb{N}$. Show that a and b are coprime.

Exercise 4
Let $N = pq$, where $p \neq q$ are primes. Assuming that the roots of a polynomial can be computed efficiently, show that given only N and $\varphi(N)$, one can compute the prime factors p and q efficiently.

Exercise 5 ($\star$)
Let (g, x, y) be the output of the Extended Euclidean Algorithm on input a, b. Show that, if $a \geq b > 0$, we have $|x| \leq b/g$ and $|y| \leq a/g$.

Exercise 6
In this exercise, you have to show how to efficiently implement a general version of the Chinese remainder theorem. More formally, prove the following. Suppose we are given relatively prime numbers $N_1, \ldots, N_t$ and numbers $a_1, \ldots, a_t$ such than $0 \leq a_i < N_i$. Moreover, let $N = \prod_i N_i$. Show that in time $O(len^2(N))$ one can compute the unique integer $a < N$ such that $a \equiv a_i \mod N_i$ for $i = 1 \ldots, t$.

Exercise 7 ($\star$)
The RSA system can be easily attacked if the public or private key is not chosen carefully. Suppose for example that the same message $m < N_1$ is encoded with the public keys $(3, N_i)$

for $i = 1,2,3$, with $N_1 < N_2 < N_3$ being relatively prime. Let b_1, b_2, b_3 be the encrypted messages. Show how to efficiently deduce m from the knowledge of b_1, b_2, b_3 and of the public keys.

Exercise 8
Implement the fast modular exponentiation function.

Exercise 9 ($\star$)
Recall the Fibonacci numbers: $F(0) = 0$, $F(1) = 1$, and $F(n) = F(n-1) + F(n-2)$ for $n \geq 2$. Consider the following two algorithms for computing the $n-$th Fibonacci number.

```
Fib_1(n):  input= n ∈ Z_+

  if n == 0 or n == 1
      return n

  return Fib_1(n-1) + Fib_1(n-2)

Fib_2(n):  input= n ∈ Z_+

  Let A = ( a  b ) = ( 1  1 )^(n-1)
          ( c  d )   ( 1  0 )

  Return a
```

Note: The computation on A is left intentionally vague. How can this be done efficiently?

a) Prove they are correct.

b) Estimate their running time.

c) Implement them and compare their running time for different values of n.