

## Fundamental Algorithms

### Midterm Exam

Exercise 1:

1. Prove or disprove  $\Theta(x^2) = \Theta(x^2 + 1)$
2. Prove or disprove  $x^n \in o(x^{n+1})$
3. Let  $n$  be an integer  $> 1$  and  $f(x) = x^n$ . For what real nonnegative values  $a$  do we get  $f(x) \in o((ax)^n)$ .
4. For what real nonnegative values of  $a$  do we get  $f(n) = \sum_{i=1}^n a^i = O(a^n)$
5. For what real nonnegative values of  $a$  do we get  $f(n) = \sum_{i=1}^n a^i \in O(1)$

Exercise 2:

Suppose  $n = 4^m$  for some positive integer  $m$ . A divide and conquer algorithm subdivides a problem into 4 sub problems of equal size. For the time complexity the equations

$$T(1) = 1 \quad \text{and}$$

$$T(n) = 4T(n/4) + f(n) \quad \text{for } n > 1$$

are given.

Give explicit formulas or, if this is too difficult for you, an upper bound for  $T(n)$  for the following cases:

1.  $f(n) = 1$  for all integer  $n > 0$
2.  $f(n) = n$  for all integer  $n > 0$
3.  $f(n) = \sqrt{n}$  for all integer  $n > 0$
- 4.

Exercise 3:

In the lecture it was shown that the average time complexity for bucket sort to sort a given set  $S$  of  $n$  elements is  $O(n)$  if the probability  $P_i$  that an element of  $S$  comes into bucket  $i$  is exactly  $1/n$  (equidistribution). Consider now the more general case

$\frac{1}{2n} \leq P_i \leq 1$  where still  $\sum_{i=1}^n P_i = 1$ . Is the time complexity still  $O(n)$ ?

Exercise 4:

In the most simple case of a divide and conquer algorithm working on a set of  $n$  elements we subdivide the set into two subsets of size  $m$  and  $n-m$ , where one of the subsets can be thrown away after having done some analysis with time complexity  $f(n) = n$ . For the (worst case) time complexity of the algorithm we may have (assume  $m \leq n/2$ )

$T(1) = 0$  and

$T(n) = T(n-m) + n$ .

Compute an upper bound for  $T(n)$  for

1.  $m = n/2$  (assume for simplicity that  $n$  is some power of 2)
2.  $m = \lceil n/4 \rceil$
3.  $m = 1$
4.  $m = \lceil \sqrt{n} \rceil$  (not so easy!).