Algorithmic Discrete Mathematics 2. Exercise Sheet

TECHNISCHE UNIVERSITÄT DARMSTADT

Department of Mathematics
PD Dr. Ulf Lorenz
Dipl.-Math. David Meffert

SS 2012
2. and 3. May 2012
Version of April 26, 2012

Groupwork

Exercise G1 (Master-Theorem)
Determine, if possible, fixed bounds for the complexities of the recurrences

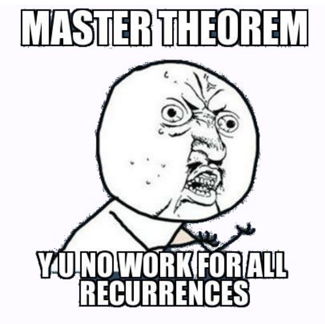
(a) $T(n) = 4T(\frac{n}{2}) + n^3$,

(b) $T(n) = 4T(\frac{n}{2}) + n$,

(c) $T(n) = 4T(\frac{n}{2}) + n^2 \log n$,

(d) $T(n) = 4T(\frac{n}{2}) + n^2$.

Hint:



Exercise G2 (Complexity)

(a) Let $f, t : \mathbb{N} \to \mathbb{R}$ be functions with $f \in O(t)$. Prove $O(f) + O(t) \subseteq O(t)$ and $O(f) + O(f) \subseteq O(t)$.

(b) Does $3^{3+n} \in O(3^n)$ hold?

(c) Does $3^{3n} \in O(3^n)$ hold?

(d) Show that $O(f) \cdot O(g) = O(f \cdot g)$ holds for $f, g : \mathbb{N} \to \mathbb{R}_+$.

Remark: For real valued functions $f, g : \mathbb{N} \to \mathbb{R}$ one just substitutes $f(n)$, $g(n)$ with $|f(n)|$, $|g(n)|$ in the definition of $O(g)$.

Exercise G3 (Algorithms)

(a) Given two algorithms A and B:

- Algorithm A has complexity $O(f)$.
- Algorithm B has complexity $O(g)$.

We want to look at two new algorithms using A and B.

Algorithm 1

INPUT : $n \in \mathbb{N}$
for $i = 1, ..., 100$ **do**
  run algorithm A
end for
for i= $1, ..., \frac{n}{2}$ **do**
  run algorithm B
end for

Algorithm 2

if $n \geq 30$ **then**
  run algorithm A
else
  run algorithmus B
end if

We already know $f \in \Omega(g)$. Determine the best possible estimates for the runtime of both algorithms.

(b) Take a look at algorithm 3 and determine the best possible estimate for its runtime. Justify you answer.

Algorithm 3

INPUT : n $\in \mathbb{N}$
m = n
while m > 1 **do**
  for j = 1,...,$\frac{n}{2}$ **do**
    a=3 · b
    c = a +b
  end for
  m = $\frac{1}{2}$· m
end while

Exercise G4 (Sets)
Order the functions

$$n^2, \sqrt{n}, n!, n^n, n$$

by their complexity. Start with lowest complexity and use the o-notation. Determine n_0 dependend on $c > 0$ in every of those cases, too.
Remark:

$$f \in o(g) :\Longleftrightarrow \ \forall c > 0 \exists n_0 \in \mathbb{N} \forall n \geq n_0 : 0 \leq f(n) < cg(n)$$

Homework

Exercise H4 (Asymptotics) (14 points)

(a) Prove that for $r_1, r_2 \in \mathbb{R}_+$ we have $n^{r_1} \in O(n^{r_2})$ and $r_1^n \in O(r_2^n)$ iff $r_1 \leq r_2$.

(b) Prove the following statements for functions $f, t : \mathbb{N} \to \mathbb{R}$:
   i. $O(f) + O(f) \subseteq O(f)$.
   ii. $O(f) \cdot O(t) \subseteq O(f \cdot t)$.
   iii. $\max\{f, t\} \in \Theta(f + t)$ for $f, t \geq 0$.

Exercise H5 (A sorting algorithm) (10 points)
The algorithm *SortList* sorts a sequence of numbers in ascending order.

Algorithm 4 SortList(*list*)

INPUT: sequence of numbers, $list = a_1, ..., a_n,\ a_i \in \mathbb{N}$
if n <=1 **then**
  return *list*
else
  $leftlist = a_1, ..., a_{\lceil \frac{n}{2} \rceil}$
  $rightlist = a_{\lceil \frac{n}{2} \rceil + 1}, ..., a_n$
  return Sort(SortList(*lelftlist*),SortList(*rightlist*))
end if

Algorithm 5 Sort(*rightlist*, *leftlist*)

INPUT: two sequences of numbers:
$rightlist = a_1, ..., a_l,\ leftlist = b_1, ..., b_k,\ a_i, b_i \in \mathbb{N}$
newlist
while *rightlist* and *leftlist* not empty **do**
  if first element of *leftlist* <= first element of *rightlist* **then**
    append first element of *leftlist* to *newlist* and delete it from *leftlist*
  else
    append first element of *rightlist* to *newlist* and delete it from *rightlist*
  end if
end while
while *leftlist* not empty **do**
  append first element of *leftlist* to *newlist* and delete it from *leftlist*
end while
while *rightlist* not empty **do**
  append first element of *rightlist* to *newlist* and delete it from *leftlist*
end while
return *newlist*

(a) Sort the sequence $9, 10, 7, 3, 1, 2, 12, 9, 23$ in ascending order by using the algorithm *SortList*. Make sure to include detailed steps for the algorithm in your solution to indicate that you understand how it works.

(b) What is the runtime of the algorithm *SortList*?

Exercise H6 (6 points)

Given algorithm 6. What does the algorithm? Determine its runtime.

Algorithm 6

INPUT : $n \in \mathbb{N}$
K1 = 2;
K2 = n;
while K2 > K1 **do**
  K2 = n/K1
  if $\lceil K2 \rceil == K2$ **then**
    return K1
  else
    K1=K1+1
  end if
end while
return 0