

## Computational Complexity – Homework 1

### Exercise 1.1

Recall the definition of the Landau notation for  $f, g : \mathbb{N} \rightarrow \mathbb{N}$ :

$$\begin{aligned} f \in \mathcal{O}(g) & :\Leftrightarrow \exists c \in (0, \infty) \exists n_0 \in \mathbb{N} \forall n > n_0 : f(n) \leq c \cdot g(n). \\ f \in \Omega(g) & :\Leftrightarrow g \in \mathcal{O}(f) \\ f \in \Theta(g) & :\Leftrightarrow f \in \mathcal{O}(g) \wedge f \in \Omega(g) \\ f \in o(g) & :\Leftrightarrow \forall \epsilon \in (0, \infty) \exists n_0 \in \mathbb{N} \forall n > n_0 : f(n) \leq \epsilon \cdot g(n) \\ f \in \omega(g) & :\Leftrightarrow g \in o(f). \end{aligned}$$

*Remark:* Some authors prefer to write  $f = \mathcal{O}(g)$  instead of  $f \in \mathcal{O}(g)$ . As  $\mathcal{O}(g)$  is set of functions, while  $f$  is a function, the latter is more precise than the former.

(a) Assume  $f, g$  are strictly positive functions, i.e.,  $f(n), g(n) > 0$  for all  $n \in \mathbb{N}$ . Show or disprove:

- $f \in \Theta(g)$  if and only if there exist  $c_1, c_2 \in (0, \infty)$  such that  $c_1 \leq f(n)/g(n) \leq c_2$  for almost all  $n \in \mathbb{N}$ . (“almost all” is equivalent to “except for finitely many”).
- $f \in o(g)$  if and only if  $\lim_{n \rightarrow \infty} f(n)/g(n) = 0$ .

(b) Let  $f$  and  $g$  be any two of the following functions. Describe their relation using the Landau notation.

$$\begin{array}{lll} (a) n^2 & (b) n^3 & (c) n^2 \log n \\ (d) 2^n & (e) n^n & (f) n^{\log n} \\ (g) 2^{2^n} & (h) 2^{2^{n+1}} & (j) n^2 \text{ if } n \text{ is odd, } 2^n \text{ otherwise.} \end{array}$$

(c) Describe (and prove) the relations between  $2^{\mathcal{O}(n)}$ ,  $\mathcal{O}(2^n)$  and  $2^{n^{\mathcal{O}(1)}}$ .

### Exercise 1.2

For  $a, b, c$  positive integers with  $c \geq 2$  show or disprove that

$$a2^{n \cdot b \cdot c^n} \in 2^{2^{\mathcal{O}(n)}}.$$

### Exercise 1.3

Consider the following language on  $\{0, 1\}$ :

$$L = \{u0v0w \in \{0, 1\}^* \mid u, v, w \in \{1\}^* \wedge |v| \leq |w| \leq |u| \wedge \exists k \in \{|v|, \dots, |w|\} : k \text{ divides } |u|\}.$$

Its characteristic function  $f_L$  is then

$$f_L : \{0, 1\}^* \rightarrow \{0, 1\} : x \mapsto \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{if } x \notin L \end{cases}$$

Construct a Turing machine which computes  $f_L$  in time  $\mathcal{O}(n^k)$  for some fixed  $k > 0$ .

### Exercise 1.4

If  $f : \{0, 1\}^* \rightarrow \{0, 1\}$  is computable by a TM with a finite alphabet  $\Gamma$  then it is also computable by a TM with alphabet  $\Sigma = \{0, 1, \square, \triangleright\}$ , moreover, with only a polynomial overhead.

Prove the statement above. Does the same hold for infinite  $\Gamma$ ? Does the same hold for  $\Sigma = \{1, \square, \triangleright\}$ ?

### Exercise 1.5

Call a Turing machine  $M$  *oblivious* if the positions of its heads at the  $i^{\text{th}}$  step of its computation on input  $x$  *depend only* on  $i$  and  $|x|$ , but not  $x$  itself.

Let  $L \in \mathbf{DTIME}(T)$  with  $T : \mathbb{N} \rightarrow \mathbb{N}$  time-constructible. Show that there is an oblivious Turing machine which decides  $L$  in time  $O(T^2)$ .

### Exercise 1.6\*

Let  $M$  be a Turing machine with a (read only) input tape and one combined work/output tape. We assume that  $M$  decides a language  $L \subseteq \{0, 1\}^*$ , i.e., every computation of  $M$  on an input  $x \in \{0, 1\}^*$  terminates eventually and after terminating the left-most position of the work tape will either be 1 if  $x \in L$  or 0 if  $x \notin L$ .

We further assume that  $M$  never writes any “blank”  $\square$ . The space  $s(x)$  used by  $M$  when processing an input  $x$  is then simply the number of non-blank symbols on the work/output tape after the computation of  $M$  on  $x$  has terminated.

- (a) A reduced configuration is defined to be any tuple we obtain from any configuration of  $M$  by forgetting about the input tape, i.e., a reduced configuration only remembers the control state and the contents and head positions of the  $k$  work tapes. Given an input  $x$ , let  $C_i(x)$  be the set of all configurations of the computation of  $M$  on  $x$  for which the input head reads the  $i^{\text{th}}$  input symbol  $x_i$ . Let  $R_i(x)$  be the set of reduced configurations we obtain from  $C_i(x)$ .

Let  $x = x_1x_2 \dots x_n$  be an input of length  $n$  such that for any input  $y$  of length at most  $n - 1$  we have  $s(y) < s(x)$ .

- Show that  $R_i(x) = R_j(x)$  for  $1 \leq i < j \leq n$  implies that  $x_i \neq x_j$ .

*Hint:* Assume that  $R_i(x) = R_j(x)$  and  $x_i = x_j$  for some  $1 \leq i < j \leq n$ . Consider then the input  $y = x_1 \dots x_i x_{j+1} \dots x_n$ , i.e., we obtain  $y$  from  $x$  by canceling the symbols on positions  $i + 1, \dots, j$ . For this input one can show that

$$R_k(y) \subseteq R_k(x) \text{ for } 1 \leq k \leq i, \text{ resp. } R_k(y) \subseteq R_{k+(j-i)}(x) \text{ for } i < k \leq n - (j - i). \text{ (Proof?)}$$

Show that this property entails the contradiction that  $M$  requires less than  $s(x)$  space for processing  $x$ .

(b) Set  $f(n) := \max\{s(x) \mid x \in \{0, 1\}^n\}$  and assume that  $f(n)$  is unbounded.

- Show that  $f(n) \notin o(\log \log n)$ .

*Hint:* Use the result of (a) to get an upper bound on  $n$  depending only on  $f(n)$ .