Computational Complexity

Homework Sheet 1

Hand in before the lecture of February 14
Preferably by email to `J.M.Czajkowski@cwi.nl`

Exercise 1 (3pt)**.** For the following pairs of functions and relations (i.e., O, o, ω, Ω, Θ), prove for the two relations at each pair whether they hold or do not hold.

1. $f(n) = 3^{(\log n)^2}$ $\quad g(n) = n^{\log n}$ $\quad$ **(a)** $g \in \Omega(f)$? $\quad$ **(b)** $f \in \Theta(g)$?

2. $f(n) = 3n$ $\quad g(n) = (\log n)^2$ $\quad$ **(a)** $g \in O(f)$? $\quad$ **(b)** $f \in \omega(g)$?

3. $f(n) = n^3$ $\quad g(n) = n^3 - 10n^2$ $\quad$ **(a)** $g \in o(f)$? $\quad$ **(b)** $f \in \Theta(g)$?

Exercise 2 (1pt)**.** Give a function $g(n)$ such that $g \in o(f)$, where $f(n) = 2n$ (and prove that this is the case).

Exercise 3 (3pt)**.** Let $\mathbb{M} = (\Gamma, Q, \delta)$ be a 2-tape Turing machine that computes some function $f : \{0,1\}^* \to \{0,1\}$ in time $t(n)$, for some function t. Give a 3-tape Turing machine $\mathbb{M}' = (\Gamma', Q', \delta')$ that computes the same function f in time $O(n) + t(n)/2$.

- Use the conventions and notation from the book (see Section 1.2 of Arora & Barak, 2009)—for example, the first tape is the input tape and is read-only.
- No need to specify $\mathbb{M}'$ in full detail; explain how $\mathbb{M}'$ is constructed.
- (*Hint:* create the alphabet Γ' in such a way that sequences $(\sigma_1, \ldots, \sigma_k)$ of symbols from Γ (of a certain length k) are encoded by a single symbol $\sigma' \in \Gamma'$.)

Exercise 4 (3pt)**.** Suppose that you are trying to determine your friend's password. You know that your friend has a password p consisting of a string of 24 hexadecimal digits, that she chose uniformly at random. Moreover, you are given a string h of 40 hexadecimal digits, that is a hash of her password p. That is, there is a function $\mathsf{hash}(\cdot)$ such that $\mathsf{hash}(p) = h$. Suppose further that there is no string p' of 24 hexadecimal digits such that $p \neq p'$ and $\mathsf{hash}(p') = h$. You also have access to an algorithm A, that given a string x of n hexadecimal digits, computes $\mathsf{hash}(x)$ in time $O(n)$.

Now consider an algorithm B that does the following. It takes no input. It iterates over all possible strings s consisting of 24 hexadecimal digits. For each such string s, it computes $\mathsf{hash}(s)$, using algorithm A as a subroutine. If it encounters a string s such that $\mathsf{hash}(s) = h$, it outputs s and terminates.

(a) Argue that the algorithm B runs in constant time, i.e., in time $O(1)$.

(b) Explain why algorithm B is unlikely to work in practice for finding your friend's password—despite its constant running time.

(c) Explain why an algorithm that runs in polynomial time—that is, in time $O(n^c)$ for some constant c—might not terminate in any practically useful period of time.

Remark 1. Answers will be graded on two criteria: they should (1) be correct and intelligent, and also (2) concise and to the point.